University of Alberta

ENHANCED RENDERING OF FLUID FIELD DATA USING SONIFICATION AND VISUALIZATION

by

Maryia Kazakevich

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2007

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-33273-3
Our file Notre référence
ISBN: 978-0-494-33273-3

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

University of Alberta

Library Release Form

Name of Author: Maryia Kazakevich

Title of Thesis: Enhanced rendering of fluid field data using Sonification and Visualization

Degree: Master of Science

Year this Degree Granted: 2007

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Maryia Kazakevich

Date: June 14, 2007

Abstract

Exploring data through various visualization techniques has become common place in science, engineering, banking, and many other domains where massive data need to be explored and analyzed by professionals to make discoveries or to take decisions. The field of scientific visualization has developed many tools that are widely used today to represent and explore complex and large multi-dimensional data sets. In many fields using visualization techniques only to explore data sets has been shown to suffer from many limitations because the representation of multi-dimensional data on a 2D screen is difficult and not always intuitive. In the scientific visualization community, this problem is called the *dimensionality curse*. Other sensing modalities, such as sound or haptics can be used either as an alternative or as a complement to visualization to transform complex multi-dimensional data set into one that is easy to perceive. This thesis describes such an interface. We show how a combination of conventional visual rendering with sound rendering can improve the localization of vortex centers in a virtual wind tunnel application. We describe the various elements of this multi-modal interface. In order to evaluate how effective this interface is, we performed a usability study that compared interfaces with sound alone, visualization alone, and visualization and sound combined. The experimental results show that the combination of visualization and sound significantly improve the usability of the interface in a vortex localization task.

To Mom and Dad,
Who always encouraged me to do my best.

Table of Contents

1 Introduction 1
- 1.1 Complex Flow Fields and Data Visualization and Sonification . . . 3
- 1.2 Multi-Modal Interface . . . 5
- 1.3 Thesis Objectives . . . 6
- 1.4 System Configuration . . . 6
  - 1.4.1 Max/MSP Unit for Real-Time Data analyzes and mapping to Sound . . . 8
- 1.5 Usability Study . . . 8
- 1.6 Thesis Structure . . . 9
- 1.7 Thesis Contributions . . . 10

2 Sound Acoustics and CFD Background 12
- 2.1 Basic Acoustical Concepts . . . 12
  - 2.1.1 Sound Representation . . . 12
  - 2.1.2 Psycho-acoustics . . . 16
- 2.2 Basics of Computational Fluid Dynamics and Fluid Flow Visualization . . . 18
  - 2.2.1 Computational Fluid Dynamics . . . 19
  - 2.2.2 Tensors . . . 20
  - 2.2.3 Fluid Flow Visualization . . . 20
- 2.3 What is Sonification? . . . 21
  - 2.3.1 Mapping Data to Sound . . . 21
  - 2.3.2 Types of Sonification Algorithms . . . 23
- 2.4 Conclusion . . . 24

3 Previous Work 26
- 3.1 Sonification Examples . . . 26
  - 3.1.1 Generic Toolboxes . . . 26
  - 3.1.2 Data Specific Algorithms . . . 28
- 3.2 Conclusion . . . 32

4 Fluid Field Sonification Algorithms 35
- 4.1 Multi-Modal Interface . . . 35
  - 4.1.1 System Architecture . . . 35
  - 4.1.2 Data Set and Solution Server . . . 36
  - 4.1.3 Haptic Interface . . . 38
  - 4.1.4 Visualization Interface . . . 39
- 4.2 Sonification Interface . . . 40
  - 4.2.1 Max/MSP for Sound Synthesis . . . 41
  - 4.2.2 Simplified Max/MSP Program . . . 43
  - 4.2.3 Data Processing Inside the `ms_client` Object . . . 45
  - 4.2.4 Data to Sound Mapping . . . 46

4.3 Types of Sonification . . . 47
4.3.1 Sound Synthesis Without Interpolation . . . 47
4.3.2 Contrast and Inverted Amplitude Modulation . . . 48
4.3.3 Frequency Modulation . . . 49
4.4 Conclusion . . . 50

5 Usability **51**
5.1 Usability Study Goals . . . 51
5.2 Experimental Protocol . . . 52
5.3 Results . . . 55
5.3.1 Sound Only vs Visual Component . . . 56
5.3.2 Visual Only vs Visual + Sound . . . 59
5.3.3 Effect of the Rendering Parameters in the Visual + Sound Condition . . . 60
5.4 Conclusion . . . 62
5.4.1 Drawbacks of the Audio-only system . . . 63
5.4.2 Advantages of The Multi-Modal Interface . . . 64
5.4.3 Audio Rendering Parameters . . . 66

6 Conclusion **68**
6.1 Summary . . . 68
6.1.1 Multi-Modal System Summary . . . 69
6.1.2 Usability Study Summary . . . 70
6.2 Future Development . . . 71
6.2.1 Improvements and Other System Setups . . . 71
6.2.2 Self-Adjustable system . . . 72
6.3 Possible Applications . . . 73
6.3.1 Virtual Wind Tunnel . . . 74
6.3.2 Real-time Architecture . . . 74
6.3.3 Wind Studies Over a Terrain Model . . . 75
6.4 Final Remarks . . . 76

Bibliography **78**

Appendix **81**

A User Studies Instructions **81**

B Measure and Log(Measure) graphs **82**
B.1 Time vs. Log(Time) . . . 82
B.2 ArcLength vs. Log(ArcLength) . . . 83
B.3 Ratio vs. Log(Ratio) . . . 84
B.4 Error vs. Log(Error) . . . 85

List of Tables

2.1 Classification Grid 24

3.1 Comparison Table 33

5.1 Sound Summary Table: effects of different rendering parameters on sound only trials (shows p values for log measure, as well as the best setup if there is an effect) 58

5.2 Multi-Modal Summary Table: effects of different rendering parameters on sound+visual trials (shows p values for measure and log measure, as well as the best setup if there is an effect) 63

List of Figures

1.1 Real-Time CFD Solver Architecture 2
1.2 Glyphs and Hyperstreamlines of a fluid flow: a simple arrow glyph representing 3D data point, a complex Haber stress glyph representing instantanious tensor quantity, and a hyperstreamline representing change in the tensor quantity 3
1.3 Arrow and sphere glyphs showing pressure and velocity values at every point 4
1.4 Multi-modal interface system diagram 5
1.5 Simplified system diagram of the interface 7

2.1 A simple sine wave 13
2.2 An example of an amplitude envelope over a sound wave 14
2.3 Representation of attack, sustain and decay in the amplitude envelope 14
2.4 White noise 15
2.5 Instantaneous amplitude probability distribution of Gaussian noise . 15
2.6 Bandpass filter 16
2.7 A complex sound wave, decomposed into three simple waves, A, B and C 17
2.8 Curves of Equal Loudness 17
2.9 2D dataset with sound fields around each data point 24

3.1 A path of the user L though a field with three sound source, s1, s2 and s3 29

4.1 System Block Diagram 36
4.2 a) Volume mesh; b) a cell; c) 2D example 37
4.3 CFD visualization of the airflow over Mount-Saint Helens with a zoomed in region showing fluid field velocity arrows, virtual pointer and local region interaction sphere 39
4.4 Max/MSP visual programming environment 42
4.5 Simple program in Max/MSP environment 43
4.6 Velocity vector, interpolated from the cell vertices, at the position of the virtual pointer 46

5.1 Trials each person had to go through for the experiments 53
5.2 Visual representation of a flow field with a high-seed vortex 54
5.3 Experimental setup used in the experiment 55
5.4 The results of trials with audio only vs trials with visual component 56
5.5 The results of wave-modulation factors on error in audio only trials . 58
5.6 The results of multi-modal interface vs. visual only interface 59
5.7 The results of wave-modulation schemes on time and error in visual + audio trials 60
5.8 Interactions of different factors in visual + audio trials 61

5.9 Multi-modal system in a specific setup vs. visual only system for error criteria 62
5.10 Sound Loudness (Y axis) vs. Square Distance (X axis) from the goal, showing minimum, maximum and average loudness values in that distance. a) Average over all 36 fields; b) One of the fields . . . 64
5.11 Error Distance vs. Field Number, showing closest to the goal local minimum and local maximum for every field a) Error distribution in the case of audio-only interface; b) Error distribution in the case of the multi-modal interface 65

6.1 Terrain model of Mount Saint Helens 75
6.2 Velocity vectors over Mount Saint Helens for a laminar wind 76

Chapter 1

Introduction

Exploring data through various visualization techniques has become common place in science, engineering, and many other domains where massive data need to be explored and analyzed by professionals to make discoveries or to take decisions. The field of scientific visualization has developed many tools that are widely used to represent and explore complex and large multi-dimensional data sets. In many of these fields using visualization techniques only to explore data sets has been shown to suffer from many limitations because the representation of multi-dimensional data on 2D screens is difficult and not always intuitive. In the scientific visualization community this problem is called the *dimensionality curse*. It is due to the fact that, as data sets become larger and have larger dimensionality, it is becoming increasingly difficult to visualize them using simple visual encoding schemes such as color, length of arrows, glyph, and others. Other sensing modalities such as sound or haptics can be used either as an alternative or as a complement to visualization to transform complex multi-dimensional data sets exploration into something that is easy to perceive.

Adding sound to data exploration increases information bandwidth reinforcing the clues given through other senses. The process of converting data into sound is known as *sonification*. Sonification can help in the recognition of features that are not obvious from other sources. For example, the visual sense is best at processing

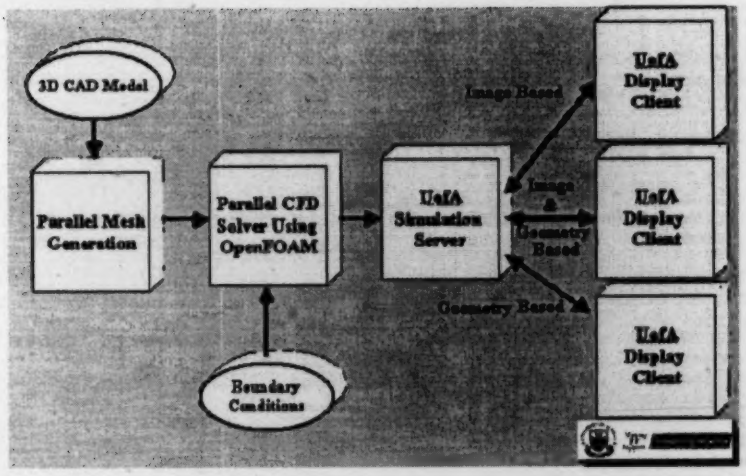


Figure 1.1: Real-Time CFD Solver Architecture

spatial information, whereas sound is better for conveying sequential or temporal information [36]. This also gives the user the possibility to process data sets using two senses instead of one. Creating such a multi-modal sensory experience is known to improve perception [36, 27]. For example, global events could be presented by sound, while local details could be presented by vision, or vice-versa. In addition, as the data sets become larger and include a larger variety of data characteristics, it might be difficult to analyze everything at once by processing only the visualization of the data. In this case, one can distribute different data attributes to different senses, reducing the effect of the dimensional curse by adding more sensory modalities. For example, pressure characteristics at each data vertex of a fluid field could be presented through visual cues, flow or vorticity through a haptic stimulation, and temperature through sound.

Sonification could be applied to any domain, but in this thesis, we will apply it mainly to Computational Fluid Dynamics (CFD) as part of a larger virtual wind tunnel project developed in the Advanced Man-Machine Interface laboratory [2]. Figure 1.1 shows a block diagram of the overall virtual wind tunnel system. In the

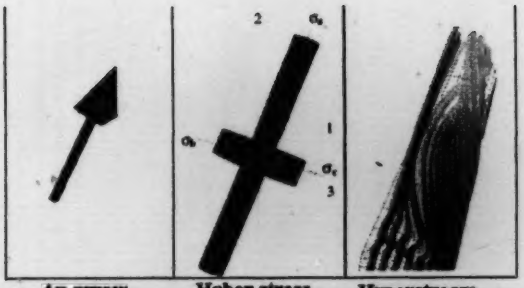


Figure 1.2: Glyphs and Hyperstreamlines of a fluid flow: a simple arrow glyph representing 3D data point, a complex Haber stress glyph representing instantanious tensor quantity, and a hyperstreamline representing change in the tensor quantity

illustrated system, a model represented as a triangulated mesh (3D CAD Model) is provided to the system and transformed through a series of steps to a model in OpenFOAM format, which is then used by OpenFOAM [21] to generate the CFD solution with the various time steps requested by the users. UofA Simulation Server controls the process, allowing various clients to connect to the simulation and share their results. The main objective of this thesis is to develop and analyze a new multi-modal interface based on visual and sound to improve the analysis of CFD data fields.

1.1 Complex Flow Fields and Data Visualization and Sonification

Fluid flow analysis requires exploring a complex tensor located on a 3-D manifold that is very difficult to interpret and communicate. This complex tensor includes pressure, speed, density, vorticity, stress tensors, and so on. A complete description of the type of data one can find in CFD, as well as tensor and glyph definitions are given in Section 2.2.

There were various attempts at tackling visualization of complex tensors, such as fluid stress tensors. Methods for complex tensor visualization include the use of

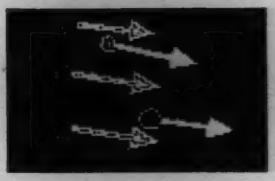


Figure 1.3: Arrow and sphere glyphs showing pressure and velocity values at every point

various types of glyphs and hyperstreamlines [13, 7]. One can see in Figure 1.2 such a glyph representation. Glyphs are geometric objects of various shape, orientation and color attributes, that are used to visualize entities in a potentially complex 3D data. Sometimes, several types of glyphs (See Figure 1.3) have to be used at once to be able to visualize all the data field properties.

There are several problems associated with these types of visualization:

- Uncertainty in glyph's placement and scale as well as in placement density;
- Uncertainty in the type of glyph to use;
- Cluttering and occlusion of information in visualization.

There is also too much emphasis on the visual sense alone. To reduce the need for visualization to represent all the information, sonification can be used. Thus, instead of loading each glyph with all the data associated with chosen point, some of the information can be presented through sound. Just like tensor components are used to describe the glyph, sound properties can be used to describe the tensor at the point of interest. There are many advantages of using sound for CFD. First, sound can be used to reinforce the information presented through visualization. Second, data properties can be split between sound and visualization to un-clutter the data presentation.

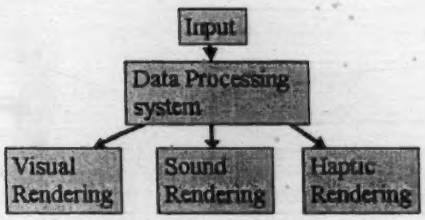


Figure 1.4: Multi-modal interface system diagram

1.2 Multi-Modal Interface

Even though one can use sound alone as a tool for analyzing data sets, it is even more useful as part of a multi-modal interface (see Figure 1.4). In the system described in this thesis, sonification is given the most attention but the other two modalities, haptic and visual, are not used to their full potentials. In a more complex system, both haptic rendering and visualization can potentially take on more interesting roles to represent various data characteristics.

In our system, the haptic interface is used for navigation within the data set and as a 3D pointing device. In a more complex setup it could also be used for rendering some of the data values in the form of force-feedback. Thus, while navigating through the field, a user could be presented with a haptic force-feedback, complex visual and sophisticated sound representation of the data, all at the same time. In this way, multiple modalities can be combined to give the system more strength in presenting the data. There are several advantages to this setup. Besides addressing the dimensionality curse, it explores the possibility of letting the user concentrate on the data rendering in the preferred modality when the data properties are presented redundantly through two or more senses.

1.3 Thesis Objectives

One of the goals of this thesis is to create a new multi-modal interface to improve the exploration of complex CFD data using sonification together with visualization. In this thesis, we are describing possible implementations and algorithms to derive useful sound rendering algorithms for a CFD data set. One of the most important tasks of incorporating sound within a rendering system is to find a mapping between data and sound. This mapping defines the main properties as well as usefulness of the sonification for a CFD application. The main focus points of this thesis are

1. What data properties are to be sonified to be meaningful for a CFD application?
2. To develop a sound rendering algorithm that is psycho-acoustically correct to enhance human perception.
3. To determine how to map CFD data characteristics to sound properties in a way that is meaningful to CFD specialists.
4. To demonstrate experimentally that the proposed mappings are meaningful and do improve the analysis of CFD data sets by users.

The last point is important since we do not want to simply create a sonification interface just for the sake of complexity, but rather to make sure it is helpful for the analysis of CFD data sets.

1.4 System Configuration

The proposed system consists of several threads that are run simultaneously to create an overall environment for visual and auditory representation of data, and to allow interaction with data through a haptic interface (see Figure 1.5).

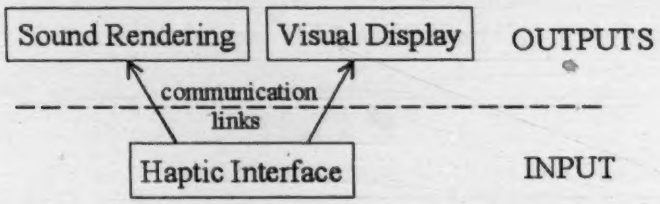


Figure 1.5: Simplified system diagram of the interface

The system was implemented on a dual Xeon PC under Windows XP operating system using Visual Studio .NET programming environment. OpenGL Performer Libraries [26] were used for visualization, while Max/MSP [6] libraries and the Max/MSP graphical programming environment were used for sonification. This allows for the sonification thread to be run on either Windows or Mac OS, while the visualization thread can be run in either Windows or Unix environments. Even though other graphical and/or visualization libraries can be used, the advantage of OpenGL Performer is that a specific visualization environment can be easily transferred to the other display systems, e.g. from the computer screen to the cave display.

Both visual and sonification threads keep the same copies of the field data. The haptic interface allows navigation through the data by means of sending haptic information to both Sound and visual programs. Both programs then simultaneously perform their own rendering of the received information to give their own data representations to the user. This divided environment allows not only for separate simultaneous calculations by both programs, preventing them to interfere with each other, but also allows for each program to be run on different threads, which is ideal from a control timing perspective. What makes this separation possible, is the use of the Virtual Reality Peripheral Network (VRPN) [1] networking library [30] for

[1]The use of the VRPN library was made possible by the NIH National Research Resource in Molecular Graphics and Microscopy at the University of North Carolina at Chapel Hill, supported by the NIH National Center for Research Resources and the NIH National Institute of Biomedical Imaging and Bioengineering

communications between the programs. It allows for all three programs (visual, auditory and haptic interfaces) to be run on different platforms, while still being able to communicate with each other, thus making the system more generic and portable.

While all three programs make up the whole system, the sonification part is of the most interest here and thus will be given most attention in further discussions in this thesis.

1.4.1 Max/MSP Unit for Real-Time Data analyzes and mapping to Sound

The main sonification thread of the whole system is mostly dependent on the Max/MSP program for digital audio rendering. The main reason for choosing Max/MSP graphical programming environment is that it has the capability for creating sophisticated sound manipulation programs, and it offers various built-in tools and objects for audio signal processing and for complex sound creation.

Another important feature of Max/MSP is that it allows custom object creation, either from already existing objects or from scratch, and then using them through a graphical interface like any other object. Even though both methods are helpful and are being used in this project, the most important one is to have the capability to write new objects from scratch and compile them into Max/MSP external objects, which can then be a part of a complex Max/MSP program. This allowed us to create a data-specific object for real-time fluid field data analysis, which constantly provides user-dependent input to the rest of Max/MSP program.

1.5 Usability Study

One of the most important parts of this thesis is to evaluate the proposed interface and specifically the proposed sonification implementations. There are several dif-

ferent types of sonification algorithms that will be discussed in detail. To compare them, a set of experiments was created that consists of three basic groups:

1. Sonification without visualization,
2. Sonification with visualization, and
3. Visualization without sonification.

The first two groups, were then further subdivided into sub-groups that include different types of sonification algorithms.

The experiments can show whether sonification is helpful at all and whether the multi-modal system is better then either sonification or visualization. They can also show which type of sonification is best, whether each sonification type works best in both multi-modal and pure sonification mode.

To evaluate each combination, users were asked to navigate through fluid fields with a given rendering (either sound or visual or both) towards a goal. The ability of a user to accurately and timely locate the goal was then evaluated and compared for each setup.

1.6 Thesis Structure

In Chapter 2, we provide some background on acoustical concepts, including discussions on basic sound concepts as well as psychoacoustics. We also provide background information on computational fluid dynamics (CFD), on sonification and data-to-sound mapping. We also present a classification system for sonification algorithms.

In Chapter 3, we discuss the relevant literature on sonification. In the discussion, we extract the most useful concepts about sonification possibilities, to evaluate what works and what does not work, and to determine how our solution fits into the relevant literature.

Chapter 4 gives the core discussion of the thesis. It presents system implementation details, the overall structure, and discusses the auditory, haptic and visualization interfaces. The major part of the discussion focuses on the sonification itself, on the Max/MSP interface and on specifics of the data-to-sound mapping algorithm. Several types of sonification are presented, which are all based on the same basic mapping algorithm.

To study which of the produced sonification algorithms gives the best result and how sonification stands in relation to visualization alone, a usability study is presented in Chapter 5. In essence, this chapter answers the question of the effectiveness of the proposed sonification algorithm and of multi-modal systems for studying complex CFD data sets.

In Chapter 6, we conclude by summarizing the project and the analysis of the result obtained by the usability study. We give an overview of the multi-modal system and the role of sonification. We also talk briefly about future developments and possible areas of application, specifically about the use of multi-modal system in a virtual wind tunnel application.

1.7 Thesis Contributions

This thesis makes contributions to research in the rendering/representation field of complex multi-dimensional data in general, as well as in sonification research in particular. This is important because sonification as a complement or replacement to visualization have long been neglected. Thus many questions about the effectiveness of sound for the perception of multi-dimensional data sets have not been addressed previously. In this thesis, we try to answer the following questions:

- What are the possible data to sound mappings for the CFD domain?
- Is sonification helpful to analyze CFD fields?

- How helpful is sonification by itself?
- How helpful is sonification to complement visualization?
- Which of the proposed mappings are the best when using sonification by itself?
- Which are the best when sonification is combined with visualization?

Chapter 2

Sound Acoustics and CFD Background

This chapter presents some basic concepts of sound analysis and CFD that are crucial for understanding the rest of the thesis, especially for understanding the the program structure and sound generation algorithms.

2.1 Basic Acoustical Concepts

At the base of sonification process is the sound itself, therefore, a solid understanding of physical and psycho-physical sound concepts is critical for creating effective sonification algorithms. Most of the acoustical concepts presented here are based on the material presented in the texts of Yost [37], Gold and Morgan [11], Gulick, Gescheider and Frisina [12], Gelfand [10], and Warren [35].

2.1.1 Sound Representation

The most basic representation of a sound wave is a simple sinusoid wave, as shown in Figure 2.1. A sound wave can be written in the form of the following equation:

$$D(t) = A\sin(2\pi ft + \theta) \tag{2.1}$$

where $D(t)$ is the instantaneous amplitude of the produced sound at time t, A is the amplitude, f represents the sound frequency, and θ is the phase of the sound wave.

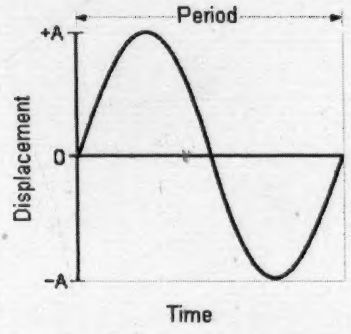


Figure 2.1: A simple sine wave

Manipulating these parameters can then lead to drastic changes in the perceived sound. In the next section, we will talk about the psycho-acoustical properties associated with these parameters.

Sound wave properties are directly dependent on vibration specifics of the sound producing object. The amplitude of these vibrations is related to the sound pressure (p), which in turn is related to the sound intensity (I). This is represented mathematically by the following equations:

$$p(t) = m|\vec{v}|/(t\ A_r) \tag{2.2}$$

$$I = p^2/\rho c \tag{2.3}$$

where $p(t)$ is the instantaneous pressure exhibited by a vibrating object at time t, m is the mass, $|\vec{v}|$ is the amplitude of the vibrating object's velocity, A_r is the area that the pressure is applied to, ρ is the density of the medium that a sound is travelling through, and c is the speed of sound in that medium.

Since the area (A_r) of a sound wave is proportional to the distance r from the sound source, the above formulas show that the sound pressure is inversely proportional to the distance (r) and that its intensity is inversely proportional to the square of the distance (r^2). This square law is a very important property of sound propagation from the sound source.

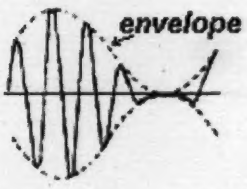


Figure 2.2: An example of an amplitude envelope over a sound wave

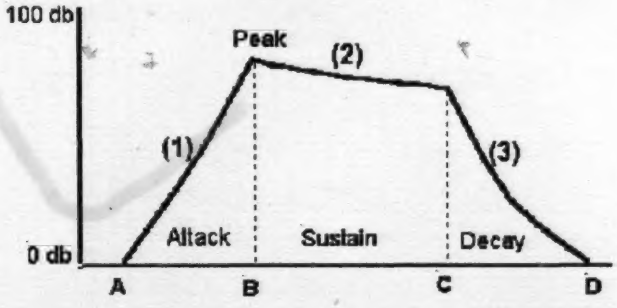


Figure 2.3: Representation of attack, sustain and decay in the amplitude envelope

Although, we can represent the simplest sounds with a sinusoidal waves, most sounds we hear in everyday life are much more complex, with either frequency or amplitude or both changing over time. For such sounds an overall relative amplitude change can be represented by an envelope (see Figure 2.2). An envelope is a maximum positive and maximum negative instantaneous amplitude displacement. An envelope can be described with attack, sustain and decay, which represent how quickly the sound reaches a maximum relative amplitude; how long it stays at a fairly constant high amplitude; and how fast the sound drops to a lower amplitude level (see Figure 2.3).

One of the complex sound waveforms that will be used in the current work is "White" noise (Figure 2.4), which can be described as a sound consisting of all the frequencies between some limits. The amplitude of every frequency varies stochastically over time, but the average amplitude of each frequency is the same. If the instantaneous amplitude varies in probability accordingly to the "normal" or

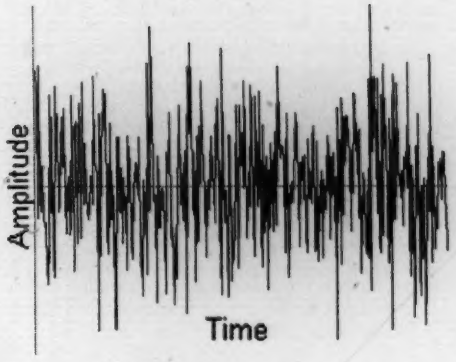


Figure 2.4: White noise

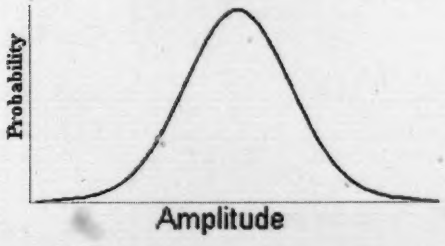


Figure 2.5: Instantaneous amplitude probability distribution of Gaussian noise

Gaussian distribution, then the noise is called "Gaussian" noise (Figure 2.5). In general, "white" noise does not have to be "Gaussian", but it is possible to have white Gaussian noise if so desired.

Any type of noise or any other complex sound wave can be modified through the use of low pass, high pass, band pass or band reject filters. As the name suggests, those filters will either amplify or reduce the amplitude of the sinusoids of certain frequencies or modify their frequency. The frequency values at which the filter starts to affect or attenuate the amplitude of the sinusoidal signal is called the cutoff frequency (Figure 2.6). The bandwidth of a bandpass filter is defined as the frequency range between the lower and upper cutoff frequencies of the filter. A filter can then be described through its quality factor (Q value) which is defined by

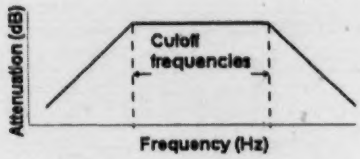


Figure 2.6: Bandpass filter

this following equation:

$$Q = \frac{\text{(central frequency of the filter)}}{\text{(bandwidth of the filter)}} \tag{2.4}$$

2.1.2 Psycho-acoustics

There are certain psycho-acoustical effects that need to be taken into account in the synthesis of sound. As mentioned in the previous subsection, the most basic sound wave characteristics are amplitude, frequency and phase. On a subjective, psychological level, these parameters are related to loudness, pitch and perceived location (in case of binaurally presented sound), respectively. The perceived location is also referred to as *pan*, which is the angle of the sound location relative to the user's head on a horizontal plane. In addition to pitch, frequency is also related to timbre. Pitch is related to the fundamental frequency of the complex sound, which is equal to the number of cycles per second in a periodic wave. A complex wave can be decomposed into simple waves (Figure 2.7). Timbre is related to the frequencies of the simple waves, that compose the bandwidth of the complex sound.

Loudness is also related to frequency. This can be easily seen by looking at the curves of equal loudness level (Figure 2.8). Each curve in this figure gives the equal loudness contour, which shows the intensity or pressure that is needed for tones of different frequencies to be perceived equally loud. As can be observed from this graph, the rate of change is different for various sections of the curves.

Another fact to consider for digital sound synthesis is the dynamic range of the human auditory system, as well as the high- and low- frequency thresholds. The

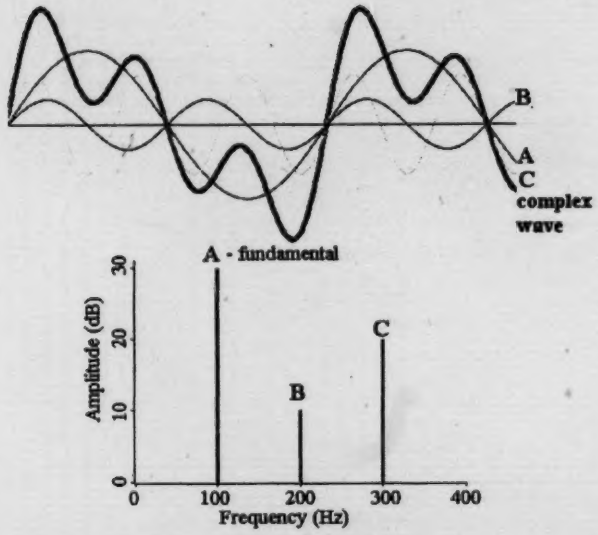


Figure 2.7: A complex sound wave, decomposed into three simple waves, A, B and C

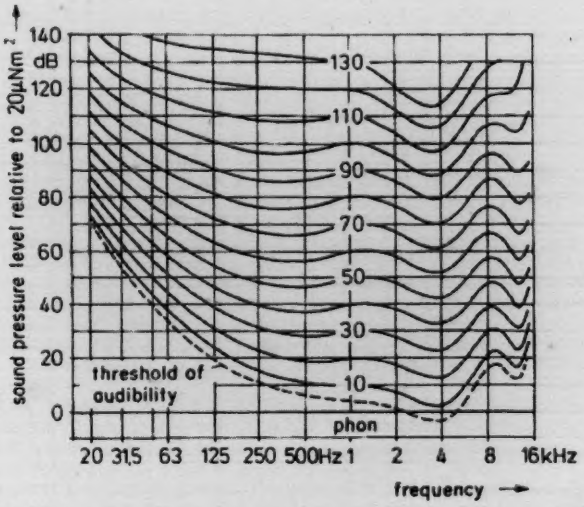


Figure 2.8: Curves of Equal Loudness

threshold of audibility is shown on Figure 2.8 for humans. Although we can hear frequencies in the range from about 20 to 20,000 Hz, it has been shown that there is some loss of sensitivity in the range outside of 1000 - 4000 Hz. Of course, individual sensitivity levels also depend on a person's age, health and possibly some other factors, but for the simplicity of the discussion we will not look at these factors.

There is a simple relationship between amplitude and perceived loudness, which shows that, even for a single frequency, the loudness of a sound is not linearly proportional to its intensity or amplitude. Therefore, we can not assume that doubling the amplitude of a sound will result in a sound being perceived to be twice as loud. The actual relationship between intensity and loudness level, as derived empirically by Stevens [28] is as follows:

$$S \propto I^{0.3} \tag{2.5}$$

where S is loudness level. Based on the Equation 2.3 then, the relationship between the pressure and loudness level will be:

$$S \propto p^{0.6} \tag{2.6}$$

This in turn means, that the relationship between the amplitude and loudness level will have the same proportionality, $S \propto A^{0.6}$.

All of the psycho-acoustical concepts discussed here influence the design of the type of sonification algorithm developed in this thesis.

2.2 Basics of Computational Fluid Dynamics and Fluid Flow Visualization

Sonification of the fluid field data requires some basic knowledge of the Computational Fluid Dynamics (CFD) concepts and techniques used.

2.2.1 Computational Fluid Dynamics

Macroscopic fluid dynamics studies the motion of molecules through control volumes by looking at the evolution of macroscopic physical fluid properties, such as pressure, velocity, density, viscosity, and temperature. Computational Fluid Dynamics (CFD) is a branch of fluid mechanics that uses numerical simulation to solve complex fluid flow processes.

At the macroscopic level, CDF is based on solving the Navier-Stokes equations, that describe the motion of fluids:

$$\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot (\rho \vec{u} \vec{u}) - \nabla \cdot \mu \nabla \vec{u} = -\nabla p + \rho \vec{g}, \tag{2.7}$$

where $\vec{u}$ is velocity vector, ρ is density, μ is viscosity and $\vec{g}$ is gravitational acceleration vector.

Usually solutions to Equation 2.7 are implemented using the Reynolds-Averaged Navier-Stokes (RANS) equations [32] or Large-Eddy Simulation (LES)[32]. RANS equations are time-averaged equations of motion of fluid flow. Here a flow variable is decomposed into time-averaged and fluctuating components. The time-averaged value of a variable is independent of time t, and is integrated over some finite, yet large time interval. In LES, large scale motions are computed directly, and small scale effects are modelled theoretically using sub-grid scale (SGS) methods [32], thus providing more accuracy and detail than RANS. LES uses spatial filtering to eliminate scales smaller than the grid spacing and requires models for the influence of the subgrid scales.

To implement either method, a fluid flow is discretized into cells to form a volume mesh corresponding to a control volume. A volume mesh is a 3D structure consisting of cells tessellating the volume and cell vertices with 3D positions. Solutions of CFD simulations provide information at every cell of the volume mesh on such fluid properties as fluid pressure, temperature, velocity, vorticity. Vorticity is a

rate of rotation of fluid velocity and can be defined as $\vec{\omega} = \nabla \times \vec{v}$, where $\vec{v}$ is fluid velocity vector. The direction of the vorticity vector is along the axis of the fluid's rotation. Vorticity is generated at fluid-solid or fluid-fluid boundaries and can not be generated internally within an incompressible fluid. A vortex describes a flow that possesses vorticity, and the vortex center is an axis around which vorticity is nonzero. Thus, a typical CFD data is composed of a complex data tensor associated with every point on a 3-D manifold.

2.2.2 Tensors

A tensor can be described as a multi-dimensional array of numbers, that transform according to certain rules under a change of coordinates. In CFD, tensors characterize the properties of the fluid field.

Complexity of the tensor first of all depends on the tensor rank. Thus, a rank zero tensor is a single number or a scalar quantity, such as temperature in the fluid field. A tensor of rank one is a vector, which is an array of dimension one, such as velocity in the fluid field. Rank two tensors are matrices or two dimensional arrays. Thus, for a rank two tensor with dimensions $< n, m >$, every point in the space would have nm values associated with it. Even with low dimension values such as $n = m = 3$, every point in space has a dimensionality of 9. Thus, for example, the behavior of a rotating body for three dimensional objects is characterizes by a 3×3 matrix, which is a 9 element two-dimensional array.

2.2.3 Fluid Flow Visualization

As was already mentioned in Section 1.1, glyphs can be used for the visualization of complex tensors. There are also other ways for tensor visualization. Thus, streamlines and their variations are intended to enhance the visualization of the vector fields [13, 7].

A streamline is a line that is tangential to the direction of the vector field at

every point along the line. A streamtube can be represented as a streamline with some thickness or a tubular region surrounded by streamlines. A streamribbon is a streamline that has been given some width and a rotation based on the local vorticity of the field.

A vortex line is a line that is tangential to the vorticity of the vector field at every point along the line. A vortex tube is also similar to the streamtube and represents a vortex line with some thickness or a tubular region surrounded by vortex lines.

A hyperstreamline is a tube with an elliptical cross section, where the parameters for the ellipse are provided by tensor values. While glyphs provide an instantaneous representation of a tensor quantity, hyperstreamlines describe the change in the tensor quantity.

2.3 What is Sonification?

Walker [34] defines sonification as the use of non-speech audio to convey information. Fundamentally sonification is the process of mapping data to sound to create a meaningful percept allowing users to understand specific properties of a signal.

2.3.1 Mapping Data to Sound

Sonification is concerned with mapping data to sound. To define a mapping we ned to answer several questions. First, do we know the meaning of a data set? If yes, are there any specific data properties that could be used to our advantage in finding a proper mapping? If not, what assumptions about the data can still be made? Second, does sonification have to be in real-time? If so, then all the calculations have to be done in real-time. If so then the data-to-sound mapping should not be too complex. Otherwise, there is still a question of whether the complex mapping will improve the understanding of the data, or whether a simple mapping is good enough, or possibly even better.

Assuming we know what the data represents, there are still various data qualities or dimensions that can be chosen to be mapped to sound. In CFD ,for example, one has pressure, velocity, temperature and other characteristics associated with each positions in the computed domain. Once one chooses data characteristics that affect the sound, there is still the question of which data points actually contribute to the synthesized sound at any time. Thus, sonification might be based on a set of interactively chosen data points in the field, on a specific path through the data set, or on all the data points belonging to a specific group of points, such as points in high-vorticity regions. Of course, there are many other possibilities, for example a path through the data set can be bound to a specific streamline, or one can choose to sonify streamribbons, streamtubes or hyperstreamlines instead of streamlines (see section 2.2). All of those have more complex characteristics, such as a twist or rotation in a streamribbon, that can be sonified.

On the other hand, there are numerous options on how to create a sound based on data values. One can chose to modify pitch, timbre, loudness, duration, spatial location, and other sound qualities. One can chose to use pre-recorded sound, fully synthesized sounds, or a mixture of both.

It is very important to decide which data characteristics affect which sound properties and in what way. For example, one can choose pressure to effect sound wave amplitude. At what rate should this effect occur? Should it follow a linear relationship? Should it have positive or negative polarity? Positive polarity is defined as a mapping, where sound value increase (rising amplitude) represents data value increase (increasing fluid velocity value). Negative polarity mapping is an inverse relationship, where sound value increase (rising amplitude) represents data value decrease (decreasing fluid velocity value). Walker [33] provides a more detailed discussion on these topics. He finds that in some cases positive polarity mapping is preferred, wheres in other cases negative polarity is better.

2.3.2 Types of Sonification Algorithms

From the previous discussions it is clear that there can be different types of sonification. Thus, whether one knows the meaning of the dataset defines how generic the produced sonification is going to be. Based on this, sonification algorithms can be divided into two basic categories, general vs. data-specific algorithms.

Generic algorithms operate on any given data regardless of what the data represents. The main idea of these algorithms is to create generic mapping tools to be used for easy sonification of any dataset. Generic algorithms are usually implemented as programmable toolboxes. A user can import his own dataset/s into such a toolbox and define how the provided data-to-sound mapping tools apply to his dataset. Basically, the user defines which values from his dataset effect which of the available sound characteristics and in what way. The mapping tools then convert chosen values from the dataset to the chosen sound characteristic through an algorithm hidden from the user. Once the user specifies all the necessary mappings, a sound is played characterizing the data.

Data-specific algorithms try to use the properties of the data set they are applied to and relate these properties to the data-to-sound mapping. In these algorithms, the data domain plays an important role. Creators of these algorithms look at what type of sound would be the most meaningful for this domain or specific dataset and what mappings would make most sense to a user.

Apart from the generic vs. data-specific distinction, sonification algorithms can be described in regards to the sound manipulation method they use. The most common ones modify such sound properties as pitch, envelope, duration, and timbre. Others modify physical properties of the sound field, such as pressure, density and particle velocity. Sound particles are created for each sound source, and then the sound field properties are modified. Each data point in the dataset can be considered a sound source with a sound field propagating from it. Figure 2.9 shows a 2D

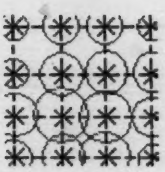


Figure 2.9: 2D dataset with sound fields around each data point

	Classification categories
1	General vs. data-specific algorithms
2	Algorithms targeted at time dependent data vs. algorithms targeted at static data
3	Different sound generation techniques, including modifying pre-recorded sound, modifying physical properties of the sound field, modifying a sound waves
4	Sonification done in real-time vs not
5	Complexity of the data-to-sound mapping

Table 2.1: Classification Grid

example where every point in the dataset is considered a sound source and sound field of each point is different. The properties of each sound field depend directly on the data values of the data point they are created for.

Modifying pre-recorded sound is also a method used. It requires for some sound, such as wind, first to be recorded from real life and then to manipulate some of its properties in accordance to the data at hand.

Classification categories of types of sonification algorithms are given in Classification Grid (see Table 2.1). There are no direct relations between either one of these classification categories. For example, if sonification belongs to the data-specific type it does not imply anything about the sound mapping complexity, or whether it works with time dependent or static data.

2.4 Conclusion

In this chapter, we presented some basic acoustic and CFD concepts and introduced the basic idea of data sonification. The main reason behind this chapter is to provide

the reader with the necessary fundamental knowledge required for understanding the rest of the discussion in this thesis.

Chapter 3

Previous Work

Several authors have attempted to use sonification for data representation and analysis. We classify these contributions in accordance with the classification grid described in Chapter 2.

3.1 Sonification Examples

In this section, we present several contributions on sonification. We divide the discussion in accordance to whether the contributions belong to the generic or the data-specific category.

3.1.1 Generic Toolboxes

General sonification toolboxes are least related to the fluid field specific sonification. However, both of the general sonification toolboxes described below use modifications of the sound waves to create sound just like our project. Another important reason for looking at them, is to determine if there is any way these toolboxes can be used in our application.

Walker's 'Sonification Sandbox' [34] takes several data sets of the same dimensionality M as input, such that the overall input is represented in the form of a $M \times N$ matrix that contains all the data. Each of the N dimensions in this data matrix is linearly mapped to either pitch, timbre, volume, or pan through a graph-

ical user interface (GUI). One of the N dimensions has to be mapped to time, and the duration that each data point is played corresponds to the relative spacing of the data points in the data set, which is the difference between two consecutive data values in that dimension. The user has a control, not only over which data sets are mapped to which sound properties, but also over how that mapping is done.

One interesting property that Walker mentions is that even though most mappings have a positive polarity, some might be better with negative polarity. Thus, an increase in the data values of one of the data parameters (such as velocity in the fluid field) does not necessarily have to be mapped to an increase in sound parameter value (such as frequency or amplitude). This is an interesting point, and we will discuss it again in the next two chapters.

The sonification produced by this toolbox is done in real-time, and the linear mapping is easy to understand. However, it is not always descriptive and meaningful; and creating sounds that are descriptive for vector flow field would require non-trivial tuning of the linear mapping functions.

Another example of a generic toolbox is Kaper's [16] work on 'Data Sonification'. Kaper uses various mapping functions for relating degrees of freedom, or dimensions, in the data set to the sound wave parameters. The sound is created through a summation of simple sine waves called partials. Static and dynamic control parameters are applied at the levels of partials as well as at the higher level of collected sound (sum of all the partials), to give more options for sound manipulation. Even though the idea is fairly straightforward, the parameter mapping functions are quite complex and require heavy off-line calculations. The created sound is not always descriptive and meaningful, and Kaper adds visualization to the sonification to help understand the sound rendering. In many ways, his work shows that if one uses several modalities together, one can improve significantly the meaningfulness of the interface for data analysis. Kaper also mentions that the

biggest problem with sonification is to find the proper mapping between data and sound, and thus sonification toolboxes should give the user flexible control over the mapping functions. However, it is that hard to define how much control should be given to the user before the toolbox becomes unusable or unhelpful.

3.1.2 Data Specific Algorithms

Data-specific algorithms are important for seeing how data specifics can be helpful for finding meaningful data-to-sound mappings. An example of a very simple data specific algorithm is the work by Noirhomme-Fraiture [20]. They create real-time sonifications of 2D and 3D time-dependent graphs, and they claim that sound is good for time series representations, because time-series, like sound, have distinguishable sequential properties. A mapping is created by relating data values to sound frequency. To create smoother sounds, outliers are first discarded and curves are pre-smoothed, thus creating easily interpretable sounds. Noirhomme-Fraiture concluded that musical and/or computer background are of only minor advantage to the users. This is important for our project given the prospect of creating a media that is useful for people with a different backgrounds.

Another example of a data-specific algorithm is Metze's [19] work on producing cell sonifications. He tries to take advantage of data properties to produce meaningful sounds that allow easy distinction between normal and malignant cell images. In this algorithm, a Fast-Fourier Transform (FFT) is first applied to the microscopic cell images. Frequency content of the FFT in then mapped to sound. Metze claims that lower frequencies predominate for malignant cells, making it possible to distinguish between the two types of cells, but they do not provide empirical support in the form of a user study. We can draw two conclusions from this work. First, if Metze's claim is true, then fairly simple mapping have potential for creating usable sonifications. Second, we can see the importance of user studies for determining

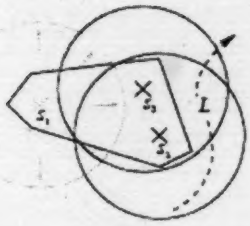


Figure 3.1: A path of the user L though a field with three sound source, s1, s2 and s3

how useful the produced sonifications are.

Ballora [1] analyzed heart rate sonifications. Here, fairly complicated mapping functions are established between the data and sound components. Data properties, like inter-beat interval characteristics, are used for creating mapping functions. The result is a set of very complex sonifications, which differ significantly between normal heart rates and abnormal ones. The paper claims that, with practice, one can train to discriminate between sonifications of various heart anomalies. Although the paper does not provide supporting evidence for this claim, it shows that complex mapping functions may in some cases be necessary for creating useful sonification.

Bovermann [3] also uses a very complicated data-to-sound mapping function, which is motivated by a physical model of heat, where local heat causes interaction between data items in space, virtually causing them to create short sounds. The user interacts with the system by virtually moving through space and causing the data items in the neighborhood of this movement to interact with each other. The mapping to sound characteristics is quite complicated and the final sound is not always meaningful. However, it demonstrates the possibility of real-time sonification, where only interactively selected data portions contributes to the sonification.

As one can see from all the examples above, most of the sonification work relies on modifying sound wave properties. A different type of sonification is presented in Shin's work [25], where a sound field is created at each data position in the data

set (see Figure 3.1). The data is given as a 3D time-varying scalar field, where physical properties of each sound field are modified according to the data values at their position. Each sound field may or may not have any affect on the final sonification. The reason for that is that while the user is navigating through the data field, a sound field at a particular position may not always reach the user at different time points on his travel path through the dataset. Sonification is not done in real time and is produced by a complicated process. The data-to-sound mapping is also complicated. However, this algorithm produces sound that is very clear, self-explanatory, and that describes the data well. An important point that we should keep in mind from Shin's work is possibility to take into account how the distance to a sound source may affect the final sonification.

A bit closer to our fluid field sonification domain is Child's work [4] on sonification of numerical fluid flow simulations. The main idea behind this sonification algorithm is to give computational fluid dynamists the option to listen to the simulation progress, so that he can easily identify if it is converging or if the solver needs to be stopped and re-started with different parameters. This way a person can have an ability to have his visual sense free for something else, while monitoring the simulation process using sonification. The main premise here is that the converging/diverging of sound should reflect convergence/ divergence of the solution process. The mapping algorithm used in this example is fairly simple and uses frequency and envelope variation. The produced sonification is not very intuitive, and contrary to its purpose, would require the user to concentrate on the sonification without being able to work on anything else. Although this might not be the best attempt at using sonification, it shows how sonification can be applied to fluid field research.

Most relevant to our sonification project is Klein's [18] work on sonification of vector fields. Data is given as a rectilinear grid of vectors. A sphere, representing

the user's head is interactively moved around the data field, and only the data samples inside the sphere are considered to reach a virtual microphone, and thus to have any affect on the produced sound wave. In this system, sonification is performed interactively and in real-time. The direction of each selected vector is mapped to the sound location, while the magnitude of the vector is mapped to level and pitch. Instead of working with simple sound waves, the parameters are applied to white nose. If all of the samples in the area are roughly of the same magnitude and direction, then a constant and smooth sound is synthesized, indicating low vorticity in the area. If vectors vary widely, sound appears to shift more, giving the impression of higher turbulence. Klein indicates that the sound produced by this algorithm is very helpful and easy to understand. However, just like other works described above, he does not provide any proof or perceptual evaluation. Klein used white noise, colored noise, pre-sampled wind noises, and cabin noise from a Boeing 747, and found white noise to be the best one. He reasoned that white noise introduced the least distortion of the data and thus was easiest to interpret. This is an important point showing that modifying synthesized sound might be better then modifying pre-recorded sound. Another important point that Klein mentions involves the size of the sampling volume used. If the sampling sphere is too small then most or all samples chosen within it will be very similar to each other thus always creating a sound indicating low vorticity even if neighboring vectors are different. On the other hand, if the sphere is too large, too many vectors might be included and the sound will always appear turbulent. This shows the significance of choosing the right size of the sampling volume.

In many ways, Klein's [18] work is very close to the current project, but there are several differences. The field he is operating on is a rectilinear grid of vectors, which implies that there are simple connections and relations between all the data points. Choosing a more complicated field grid introduces a more complicated

relationship between user position and the data points within the interaction region. Furthermore, Klein is only concerned with finding a good representation of the data in the selected region and does not allow for the sphere of influence to be modified. This makes it difficult to test the system in perceptual studies on the importance of the influence sphere size. To solve this problem one could give the user control over the influence sphere diameter, which could vary from a single point to the whole field. This system would give the user greater control over the produced sound. Alternatively one can perform a usability study to determine the optimal radius for a given data set.

3.2 Conclusion

A summary on data-to-sound mapping, its complexity and usefulness of the produced sound for each of the sonification projects described previously is given in Table 3.1. As one can see, it is not easy to create a real-time sonification that would always be meaningful. Complexity of the presented data-to-sound mapping algorithms vary significantly, however even complex mappings do not always result in a meaningful sonification. We can see several projects that use frequency modulation and other sound wave characteristics to produce sound. Most of them use a simple mapping between data and sound, but not many of them produce sonifications that are simple and easy to interpret.

The main problem with both general toolbox examples is that one is limited to the capabilities they provide. Many sonification projects bring out some useful ideas about the sound, its physical properties and mapping functions. Some of the sonification projects provide a user with possibility to only listen to the produced sonification without the ability to navigate to a specific part of the data set to listen to [34, 16, 20, 19, 1, 4]. For sonification of fluid field data, it is important to be able to explore the data set interactively, rather than just to passively listen to it. This

Name / comparison criteria	*Mappings to*	*Mapping Complexity*	*Understanding produced sound*	*User navigation*	*Real-time*
Sonification Sandbox (B.N. Walker [34])	Pitch, timbre, volume...	Very simple	Easy, but not always descriptive	None	Yes
Data Sonification and Sound Visualization (H.G. Kaper [16])	Sound wave parameters on a level of partials and collected sound	Simple idea, not very simple mappings	Not very obvious, not clear if helpful	None	No
Sonification of time dependent data (M. Noirhomme-Fraiture [20])	Frequency	Very simple	Easy, not clear if helpful	None	No
Cell Music (K. Metze [19])	Amplitude, frequency and duration	Fairly simple	Fairly helpful	None	No
Heart Rate Sonification (M. Ballora [1])	Various sound wave characteristics	Quite complicated	Very distinguishable but complicated sound	None	Yes
Local Heat Exploration Model For Interactive Sonification (T. Bovermann [3])	Characteristics of the sound grains	Very complicated	Neither clear nor obvious	Interactive	Yes
Vortex Sound Synthesis (Y. Shin [25])	Density and particle velocity	Quite complicated	Very clear and self-explanatory	Off line	No
Sonification of Numerical Fluid Flow Simulations (E. Childs [4])	Frequency and envelope	Fairly simple	Not obvious	None	Yes
Sonification of Vector Fields (E. Klein [18])	Sound location, level and pitch	Fairly simple	Helpful and easy	Interactive	Yes

Table 3.1: Comparison Table

possibility helps to involve the user within the simulation, making him not only a passive observer but an active participant, who is interacting with the sonified environment.

The sonification project described in this thesis belongs to the real time, data-specific sonification produced through the means of modifying sound wave properties. In the next chapter, we will look more closely at defining a data-to-sound mapping function between the fluid field data and a sound wave. Our hope is to create an easy-to-understand set of audio stimuli that can be used by fluid dynamists. The survey of the related literature in this chapter provided us with a start point for our project.

Chapter 4

Fluid Field Sonification Algorithms

This chapter provides a full description our fluid field sonification system. Special attention is given to the sonification algorithm itself. Discussions will also concentrate on the data processing done in preparation for the data-to-sound mapping.

4.1 Multi-Modal Interface

This section presents the main implementation details of the multi-modal interface system. It describes how the data is represented, how it is visualized, it describes the various interaction modalities, how the haptic interface works, and finally shows how the connections between the main parts of the system are assembled.

4.1.1 System Architecture

There are three threads that run simultaneously in the system (Figure 4.1). One of the threads is a part of MAX/MSP sonification program that produces sound rendering of the data. Another one is visualization thread that produces visual rendering of the data. The third one is a haptic thread that is resposible for interaction with a user through a haptic device. Both sonification and visualization threads are completely independent of each other and depend on the haptic thread to receive position and orientation of the virtual pointer as well as the radius of interaction.

Since the main purpose of this interface is to sonify CFD data, the sonification

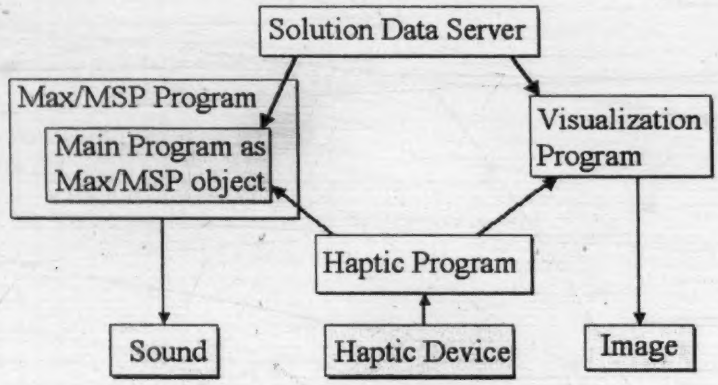


Figure 4.1: System Block Diagram

thread depends only on receiving a pointer position provided by the haptic thread. The visualization thread also depends on the haptic thread for visual interaction with the CFD data to locate in 3-D space the virtual pointer. However, if other threads are not running, the visualization thread can be run by itself without interaction. In this mode of operation, the fluid field can be rotated to be viewed from different viewpoints without interactive virtual pointer or sonification.

4.1.2 Data Set and Solution Server

Both, sonification and visualization threads connect to the simulation solution server to receive the data set. The simulation server was developed by Boulanger et al. [2]. The system uses the OpenFOAM (Field Operation and Manipulation) [21] software package to simulate complex fluid flows and the VTK library [17] to process the data field geometry. The simulation server is in charge of distributing, in real-time, the CFD time-steps computed on a remote high performance computer. Since this server is responsible for all the necessary CFD processing, we did not have to be concerned with creating a proper fluid field, and were able to concentrate only on the rendering. Connection to the simulation server is done through the Quanta libraries [8], an advanced communication package used for data exchange in VR en-

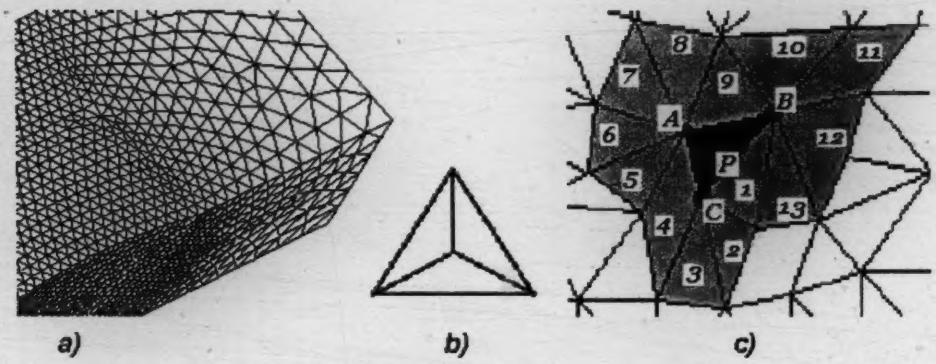


Figure 4.2: a) Volume mesh; b) a cell; c) 2D example

vironments. Sonification and visualization programs connect to the server independently. Once the data is received, a corresponding rendering process is performed to create the visual and sound displays.

The data is produced by the OpenFoam simulator [2] as a series of time-steps, with the data structure (the volumetric grid cell) staying the same but where the attribute values changing at each vertex. The term *data structure* here refers to the coordinates and connections between the cells and vertices of CFD mesh. The values at each grid cell are attributed by velocity, pressure, density or temperature values. For each time-step, any of these values can change in time as the CFD simulator progresses through each iteration.

The data set does not have a uniform structural form, such as, for an example, a rectilinear grid. Therefore, each cell and node has to keep information about all the other cells and nodes it is connected to. An example of a volume mesh is shown in Figure 4.2a. In this case, each cell is a tetrahedron, as shown in Figure 4.2b. The walls, as well as the nodes, of the cells are shared between neighboring cells. The size and density of cells change from one region to another. To deal with this non-uniformity, a commonly used data representation structure was implemented [22]. Every cell keeps track of all the nodes that belong to it, and each node keeps

track of all the cells it belongs to. This set-up helps to track the location of the virtual pointer when it is moving through the field. In order to find which cell the pointer is moving through at a particular moment, the program has to look only at the neighboring cells of the last cell the pointer was in. Figure 4.2c, shows a simplified 2D case of this set-up. When a pointer P is moving out of the central cell (described by ABC triangle), the program looks up all 13 neighboring cells, and these are the only ones it has to look at. Data sets used for this prototype were fairly large in size, having on the order of 70′000 cells. However, the interactions are local to some area of the field and a linked data structure representation provides a very fast method for performing cell traversal of this area.

4.1.3 Haptic Interface

The haptic thread is mainly responsible for receiving feedback from the user by tracking the movements that the user makes with the haptic device in 3–D. We use a PHANTOM Omni [23] as the haptic interaction device.

The haptic thread receives stylus position and orientation from the haptic device. Since these values are given in haptic device coordinate system, the thread then converts this information to the given fluid field coordinate system and sends the converted values to the visualization and sonification threads. Thus, the haptic thread is used to provide information on the position and orientation of the virtual pointer within the fluid field data, allowing users to navigate in 3–D through the fluid field.

In addition to the device position and orientation, the haptic thread also reads button information of the device. Two buttons on the device are used to interactively change the diameter of the interaction sphere (local region). The center of the local region sphere is located at the tail of the virtual pointer. The points inside the sphere are the ones affecting the sound wave produced by the sonification thread.

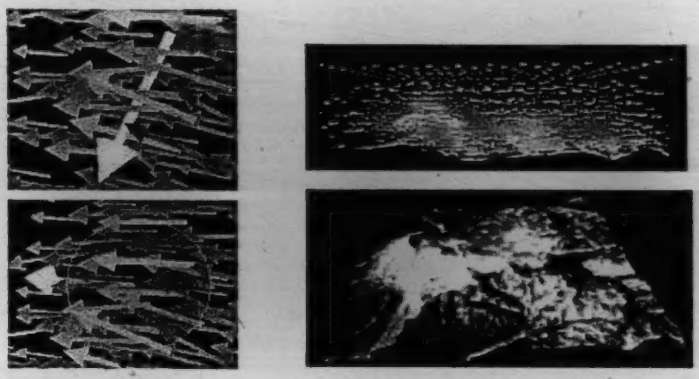


Figure 4.3: CFD visualization of the airflow over Mount-Saint Helens with a zoomed in region showing fluid field velocity arrows, virtual pointer and local region interaction sphere

A detailed description of this will be presented when we describe the sonification algorithm.

In addition to navigation, the haptic interface can also be used to provide a force-feedback to the user. In order to guide the user movements to stay inside the field boundaries, the haptic interface provides a force-feedback, disallowing movements outside of the data field boundary.

The haptic device could also be used as an even more interactive force-feedback interface, where the haptic device could give force-feedback that is proportional to the flow density and direction. In this way, the multi-modal system could provide haptic cues on the properties of the field, in addition to visual and auditory cues.

4.1.4 Visualization Interface

Because sonification is the main emphasis of this thesis, only a basic visualization interface was developed. As mentioned previously, the visualization thread receives the CFD data set from the simulation server. It requires a connection to the haptic thread to receive information on the location and orientation of the virtual pointer where the user wants to explore the data set. The visualization thread is responsible

for displaying arrows at each node of the fluid field. It also shows a representation of the virtual pointer and the local interaction area in the field (see Figure 4.3). Direction of each arrow corresponds to the velocity direction, while size and color to the velocity amplitude. A virtual pointer is displayed as a white arrow whose position and orientation correspond to the position and orientation of the haptic device relative to the flow field. The local area is represented as a sphere located at the tail of the virtual pointer, with a interaction diameter specified by the haptic thread.

As shown in Figure 4.3, the visualization thread can also display a mesh of the object, such as of the Mount-Saint Helens surface including mesh and texture in this case. This option is possible if the data file with 3D object of the mesh is given and read into the program. 3D object is represented as geometry with connected faces and possible vertex or face colors or texture coordinates. Displayed in Figure 4.3 3D terrain model of the Mount-Saint Helens, located at the United States with coordinates of 46°11′28″ N and 122°11′39″ W, was acquired at a high resolution by the Shuttle Radar Topography Mission (SRTM) [31, 9]. The mesh display can be interactively switched *on* and *off* during the run of the program. Further, using the mouse, the whole scene can be rotated to be observed from different viewpoints, as well as zoomed in and out.

4.2 Sonification Interface

Just like the visualization thread, the sonification thread receives, at each time step, the CFD data set from the simulation server, as well as the updated virtual pointer position from the haptic thread. The visual program does not have to compute anything as the pointer is moving through the cells. On the other hand, the sonification thread needs to compute and interpolate local velocity values around the virtual pointer using. Once these values are computed, they are mapped to the sound pa-

rameters creating a complex sound wave. The mapping is performed inside the Max/MSP visual programming environment (see Section 4.2.1). Complex sound waves are synthesized from a band-limited white noise where the central frequency and average intensity are modulated by the interpolated flow velocity. The resulting output signal is a sound similar to wind.

One of the reasons for using a complex sound instead of a modulated simple sinusoidal wave, is that the sound is more pleasant to hear. The reason for using a wind-like sound is that it is more intuitive with the fluid field data. Of course, white noise is not the only possible solution to the CFD sonification problem. The set of possible mappings is endless, and we chose a setup most suitable for our project. Even with the chosen setup, there are several possible mapping variations that will be discussed in the last section of this chapter.

4.2.1 Max/MSP for Sound Synthesis

As mentioned in section 1.4.1, Cycling '74's Max/MSP [6] programming environment offers many capabilities for sophisticated digital sound processing. It is one of the most popular packages for electronic music composition. It provides an easy interface for graphical, stream-based programming to process streams of audio and MIDI data.

Max started of as a project for producing interactive music, and over the years it evolved into a commercial program. It provides a user with a high level graphical programming language, such that the programs are created through manipulation of graphical objects rather than through low level programming. This feature enables many music enthusiasts, who do not have much programming skills, to be able to use Max for composing music and for sound manipulation. In addition, Max programs run in real time. Since the Max language is written in C, the users have an option of creating their own objects in C to further expand the Max functionality.

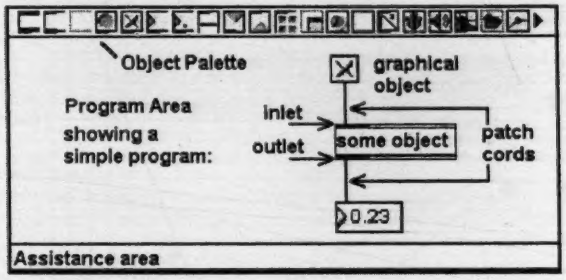


Figure 4.4: Max/MSP visual programming environment

The basic Max environment includes MIDI, control, user interface, and timing objects. It is often used in conjunction with other Cycling '74's creations built on top of it, such as, for example, Max Signal Processing (MSP), which includes a large set of audio signal processing objects.

The modular programming paradigm of the Max/MSP environment allows for fast program creation from pre-existing data processing objects provided in the graphical interface. The basic elements of Max/MSP are data processing elements, called objects, and patch cords that connect these objects. A Max/MSP interface with a simple program is shown in Figure 4.4. The object palette at the top of the interface provides the user with many predefined objects, which can be dragged to the program area and connected together with patch cords to form a program. In the sample program shown, the outlet of the first object is connected to the inlet of the second object using one patch cord and the outlet of the second object is connected to the inlet of the third object using another patch cord. Patch cords ensure the data flow in the program, while the objects do all the necessary data processing.

One of the examples, of a built-in object that we are using, is `noise~`, that generates a signal consisting of uniformly distributed random, white noise values between -1 and 1. A produced signal from that object is then further manipulated in the system by propagating it through other objects or filters. We can add several different objects to the program and connect them together to produce an audio

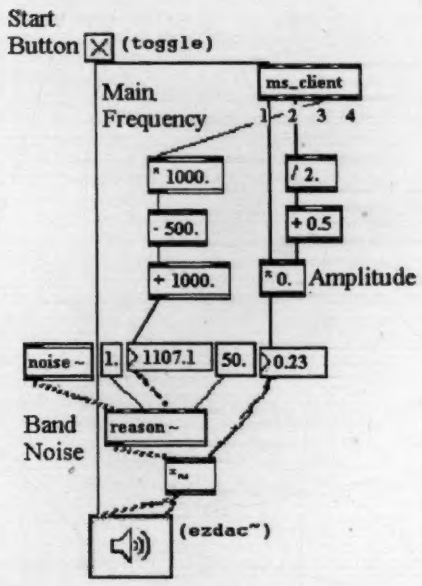


Figure 4.5: Simple program in Max/MSP environment

signal of any complexity.

As mentioned previously, Max/MSP is a visual programming environment. However, users can add customized objects called externals by writing them in C and compiling them into Max objects. These external objects can then be used in Max/MSP interface like any other Max/MSP objects. In our program, we use an external object `ms_client`, which is used for flow data manipulation based on haptic movements.

4.2.2 Simplified Max/MSP Program

A network for simple sonification in Max/MSP is shown in Figure 4.5. It shows five main objects connected to each other via patch cords: `toggle`, `ms_client`, `noise~`, `reason~`, `*~` and `ezdac~`. There are also some simple objects that perform basic calculations, such as addition, substruction, division and multiplication. Only the main objects are discussed below.

A `toggle` object sends an on/off signal to other objects causing them to start or to stop producing an output. Thus, when an 'on' signal is sent to the `ezdac~`, it sends out the sound wave signals received at its right and left inlets to the corresponding sound channels.

The `ms_client` object is responsible for most of the processing that calculates data values for the data-to-sound mapping. The calculations performed by this object will be presented in more detail in the next subsection. For now, we just need to know what the output values represent. The first and third outputs are related to the velocity value at the position of the virtual pointer. The second and forth outputs are related to the orientation of the virtual pointer relative to the flow direction at that position. All values are scaled in a range between 0 and 1. The first two outputs are calculated for amplitude mapping, while the other two are used to modulate frequency.

The next important object is the `reason~` object. It is a bandpass filter that modifies the white noise sound, which is given through the first input from the `noise~` object. The second input to the `reason~` object is gain, the third one is central frequency and the fourth one is the Q value of the filter (defined as the center frequency divided by the bandwidth). The `reason~` object knows which frequencies to filter out of the signal. In the current setup, the Q value is set to a number that will create a wind-like broadband noise. Central frequency ranges from 500 to 1500 Hz. While the frequency of the broadband noise is varied through the `reason~` object, the amplitude of the output signal is modified through a `*~` object.

In this setup, the central frequency is only mapped to the velocity values, indicating higher pitch sound and thus faster winds for faster flow. The amplitude is mapped to velocity and relative orientation values. Thus faster flow is not only louder, but also the more virtual pointer is oriented towards the flow the louder the

sound will be. A re-mapping of the relative orientation value to an interval between [0.5, 1] ensures that we hear the simulated wind even if we are not facing it.

4.2.3 Data Processing Inside the `ms_client` Object

As described previously, the `ms_client` object receives the data set from the simulation server and stores it locally, as described in Section 4.1.2. At each time step, which is set to a fraction of a second, it receives an updated location and orientation of the virtual pointer, as well as the local area radius from the haptic thread. Letting the size of the local area to be changed interactively allows for better control over the extent of the sensitivity of the virtual microphone.

Depending on the virtual pointer position $p_c = (x_c, y_c, z_c)^T$, the program determines which mesh cell the virtual pointer is in. Once the cell is found, a Shepards [24] interpolation scheme is used to interpolate the flow velocity vector $\mathbf{v}_c = (v_x, v_y, v_z)^T$ at the pointer's position based on the following equation:

$$\mathbf{v}_c = \frac{\sum_{For\,All\,Vertices} \mathbf{v}_i / \|p_i - p_c\|^2}{\sum_{For\,All\,Vertices} 1 / \|p_i - p_c\|^2} \tag{4.1}$$

Based on the law of sound propagation, the further a node is from the pointer, the less influence it has on the interpolated value. This equation is applicable if the interpolated velocity vector at the virtual pointer is calculated from the nodes of the immediate cell, or if it is calculated from all the nodes inside the local area sphere. The first case is applicable when the size of the local area is equal to or smaller than the size of the cell that the pointer is in. A graphical example of this interpolation case is shown at Figure 4.6. As shown here, the angle α between the interpolated velocity vector and the virtual pointer vector is also computed.

A more correct physically-constrained interpolation of fluid flow fields is given by Zhong et al. [38]. Their interpolation method is more complex and might be necessary for proper simulation and visualization of data. For sonification pur-

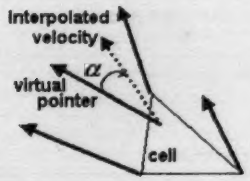


Figure 4.6: Velocity vector, interpolated from the cell vertices, at the position of the virtual pointer

poses, however, this introduces an unnecessary complexity in calculations, without noticeably changing the produced sound. For our purposes, Schaeffer's interpolation provides sufficient information for the average velocity and angle values.

Both velocity value $v = \|\mathbf{v}_c\|$ and angle value α are normalized to a value between [0,1]. These normalized values can then be mapped to a sound frequency in the Max/MSP network, similar to the simple case shown in Figure 4.5. For the mapping to amplitude, these values undergo a non-linear transformation based on psychophysical considerations, as described in Section 2.1.2, equation 2.5.

4.2.4 Data to Sound Mapping

As mentioned in Section 2.1.2, the relationship between loudness S and amplitude A can be given as follows:

$$S \propto A^{3/5} \tag{4.2}$$

If one wants to have a linear relationship between the loudness and data values, $S \propto k$, where k is the data value, the function between data values and amplitude should be:

$$A \propto k^{5/3} \tag{4.3}$$

Then the result would be: $S \propto A^{3/5} \propto (k^{5/3})^{3/5} = k$ or simply $S \propto k$.

Thus, in the simple network (Figure 4.5) amplitude $A = v^{5/3} * \alpha^{5/3}$, where both v and α values are normalized to a value in the interval [0,1]. Similarly, $\alpha^{5/3}$ is also mapped between the interval [0.5, 1] to preserve some sound even if the virtual

microphone is not facing the flow. The normalized v value is mapped to a frequency value f in the interval [500Hz, 1500Hz].

The frequency interval [500Hz, 1500Hz] was chosen for the following reasons. First, there is a bigger dynamic range of the human auditory system for these frequencies. Second, equal loudness curves vary little in this frequency range, dependency of loudness on frequency was minimized. Third, there a reduced sensitivity below 1000Hz which implies that lower frequency values are not desirable. Lastly, variations in data depended more on amplitude variation, and thus frequency did not vary as much.

Several data-to-sound mappings were used in our sonification system. In one of these mappings, frequency was set to play a much bigger role than amplitude and the frequency range was changed to [1000Hz, 4000Hz].

4.3 Types of Sonification

Various mapping configurations are possible even when using only velocity values and angle of interaction on the data side and mapping them to frequency and amplitude of broadband noise. First, one can create various sonification configurations in the Max/MSP program. Second, a more generic `ms_client` object than the one described previously was created to allow for more flexibility.

4.3.1 Sound Synthesis Without Interpolation

In the program setup described previously, the sound is produced after interpolation. This means that the interpolated value at the position of the virtual pointer is calculated first and then mapped to one sound wave characterizing all the data points in the local area around the virtual pointer.

Another way for producing sound is to view all of the vertices in the local area around the pointer as virtual sound generators located at some distance from a vir-

tual microphone. In this setup, the sound is created before the interpolation, as a separate sound for each vertex in the local area. If one applies the law of sound propagation in this setup, the distance of each vertex from the virtual pointer affects the loudness of the sound. As in the after interpolation algorithm, the radius of the local area can modified to change the extent of sensitivity of the virtual microphone. With a correctly chosen radius, the areas of the field with high velocity will produce a highly varying set of sound waves with many different amplitude and frequency values.

To implement this algorithm in MAX/MSP, a more generic `ms_client` object was created, which varies the number of output velocity and angle values in accordance with the number of vertices in the local area. A set of velocity and angle values for each node is then mapped to a separate sound wave, which uses the same bandpass filter network as the simple program in Figure 4.5.

This `ms_client` object has an input option, that can be set from within the Max/MSP network and run either before or after interpolation. A more complicated Max/MSP program had to be created to accommodate this more complex `ms_client` object and the possibility of synthesizing a variable number of sound waves.

4.3.2 Contrast and Inverted Amplitude Modulation

So far, an increase in velocity value was mapped to an increase in amplitude of the sound. This can be described as contrast amplitude modulation and positive polarity mapping (see Section 2.3.1). One can describe this algorithm as the most intuitive mapping for CFD data. Walker [33] presents a more detailed discussion on this topic. He finds that there are numerous opinions as to what constitutes the most intuitive polarity mapping for a specific dataset. Furthermore, he finds that, in many cases, although one might prefer a certain polarity mapping, a different

(non-preferred) mapping might actually be better suited for analyzing a given data set.

To investigate the importance of polarity for CFD data sonification, the `ms_client` object has the option to be run in either positive or negative polarity mode for amplitude mapping. Positive polarity mapping translates higher velocity values into a louder sound. On the other hand, negative polarity mapping gives a perception of inverted amplitude modulation, where higher velocity values result in a quieter sound, and vertices with the largest velocity values produce no sound at all. Both contrast and inverted amplitude mode can run before or afterinterpolation setup.

4.3.3 Frequency Modulation

Instead of having amplitude as the main sound modulating parameter, one can concentrate on frequency modulation. Here, velocity values v are mapped to frequency f and amplitude A is modulated by angle values α. Amplitude was set to vary in the range [0.5, 1] based on a similar formula as before, namely $A = \alpha^{5/3}$. Frequency was set to vary in the range [1000Hz, 4000Hz] based on the linear velocity to frequency mapping, as before. A Q value of the bandpass filter was set to a much larger value, resulting in a very narrow band noise.

Even though the mapping functions did not change much, the change in the frequency range and in the band of the output noise made a big change in the perceived pitch of the output sound. Now the highest pitch noise is identifiable with higher velocity values in the CFD data, without being overshadowed by simultaneous amplitude rise. This setup also helps to unload the amplitude mapping, which now represent only the flow direction of the field relative to the virtual pointer orientation. Just as with an amplitude modulation, either a positive or negative polarity is possible for frequency mapping.

To be able to look at frequency versus amplitude mapping for CFD data, an

`ms_client` object has an option to be run in frequency modulation mode instead of amplitude modulation mode. Furthermore, frequency mode can also be run with either before or after interpolation.

4.4 Conclusion

In this chapter, several data-to-sound mapping algorithms were presented. In every setup, velocity values and orientation relative to the virtual pointer were mapped to central frequency and amplitude of a broadband noise. The exact mapping functions were motivated by psycho-physical considerations. Several setup choices were possible: Do we produce sound before or after interpolation? Do we use contrast or inverted data-to-sound mapping? Do we use amplitude or frequency modulation? Any combination of the three setups can be used.

Although one setup may be more intuitive than another, a perceptual analysis is necessary to determine the efficiency of each mapping function. Furthermore we need to consider the radius of influence for informative navigation within the field. This too can be investigated through perceptual analysis in order to show how much effect a size change has, and also to determine the values that work the best. The choice of the local sphere of influence may also depend on the particular setup used. The analysis of the choice of mapping and the radius of the sphere of influence will be presented in the next chapter.

Chapter 5

Usability

Any man-machine interface needs to be tested in order to determine its effectiveness and usability. In this Chapter, we describe a usability study that was performed for a vortex localization task. This task was chosen as a representative of a typical data query operation that a CFD expert would do in the virtual wind-tunnel in order to explore the flow around wings and vehicles. This is important because the reduction of drag created by turbulence is a key design issue in automotive and aerospace design.

5.1 Usability Study Goals

The general goal of the user study presented here is to examine how one can improve the localization of vortex centers using a combination of auditory and visual feedbacks. The specific goal of this study is to determine how well auditory feedback improves visual feedback in a multi-modal interface. A comparison was performed between a pure visual feedback interface, a pure audio feedback interface and multi-modal visual/audio interface. We hypothesized that the multi-modal interface would provide the best results for all of the measuring criteria. A secondary goal of this user study was to determine which specific audio rendering algorithm and parameters would give the best results to improve localization.

As mentioned previously, we are also interested in the effect the local radius R

of influence (see Section 4.2.3) has on the localization of the vortex center. Specifically, we wanted to know whether there is an optimal size of the radius for a specific data set, and if the optimal size was the same for both auditory feedback and multimodal interfaces, and if there was an optimal radius size for a specific interface setup?

5.2 Experimental Protocol

To evaluate the system, each participant, was presented with a set of fluid fields with random vortex locations and was asked to locate the vortex center of each field. The goal for the user was to locate the node with the highest velocity value in each of the fluid fields. This node was marked with a large red arrow, or a specific sound, or both. There was always only one red arrow in the field, and it was always associated with either the loudest, the most silent, or highest pitch sound, depending on the rendering algorithm used.

Every participant was given the same number of visual, auditory and visual+auditory trials. Within the last two types of trial, there was an equal number of trials with sound produced either before or after interpolation. In addition, there was an equal number of trials with different radii of influence. To simplify interaction with the interface, the participant did not have control over the radius of influence. Rather, the interface simply set the radius to one of three different sizes, a small radius ($R = 0$), a medium radius ($R = 1.1$), or a larger radius ($R = 1.6$). Schematically, the set of experimental conditions that a participant was presented is illustrated in Figure 5.1. each participant was given six visual, six auditory and six visual/auditory trials. The trials with an audio component were then further subdivided into three trials before interpolation and three trials after interpolation, for each radius size.

The only other comparison dimensions that are missing in Figure 5.1 are whether amplitude or frequency modulation was used, and if a amplitude modulation was

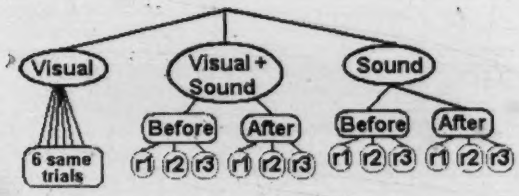


Figure 5.1: Trials each person had to go through for the experiments

used, weather contrast or inverted sound (see Section 4.3.2) was used. Each participant was given only one of the three types of wave modulation, frequency modulation, positive amplitude modulation (contrasted sound) or negative amplitude modulation (inverted sound). In the case of positive amplitude modulation, one could search for the loudest point in the field, with negative amplitude modulation, one could search for the quietest place in the field, and with frequency modulation one could search for the highest- pitch sound.

Since participants were divided in three groups, three different sets of instructions were created, one for those participating in the frequency modulation condition, one for those participating in the positive amplitude modulation condition and one those participating in the negative amplitude modulation condition. The only difference between the instructions was that the participant was made aware of the rendering scheme. A sample instruction is provided in Appendix A.

A set of fluid fields was synthesized consisting of 40 fields with the same dimensions, with 24x24x24 nodes, but with the point of interest (highest-velocity flow) located at different locations. An example of one of the fields, as presented through the visual interface, is shown in Figure 5.2.

Each participant was tested for color-deficiencies using a set of Ishihara [15] plates, since that might have compromised the results of trials with the visual component. Each person was given a set of instructions (example in Appendix A) to read before the experiment.

The experiment was divided into two parts: 15 warm-up trials and 36 experi-

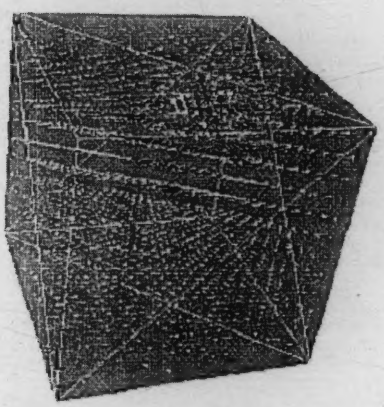


Figure 5.2: Visual representation of a flow field with a high-seed vortex

mental trials. During the warm-up trials, participants were encouraged to play with the interface and to learn the basic navigation tools of the visualization interface. The warm-up trials started with 3 visual trials to allow the user to learn the visual interface. This was followed by 12 additional trials for all 12 possible sound and visual+sound conditions, as shown in Figure 5.1. The results of the warm-up trials were not included in the analysis. The warm-up trials were followed by the experimental trials, with each of the conditions in Figure 5.1 presented twice.

In each trial, participants were presented with a randomly chosen field and randomly chosen modality. The modality was either visual only, sound only or visual + sound. This was the only information given to the participant for any trial. The participant had no information on the type of data-to-sound mapping algorithm that was used and was not made aware of the value chosen for the local radius of influence.

An image of the experimental setup is given in Figure 5.3. Participants interacted with the interface through a mouse and a haptic device. A visual rendering of the field, if given, was shown on the main monitor. The secondary monitor was used for messages to indicate whether the current trial was visual, sound, or visual

Figure 5.3: Experimental setup used in the experiment

+ sound. The audio rendering of the field was presented through headphones.

To start a trial, the participant had to press the white button on the stylus of the haptic device. Each trial started with the haptic pointer being set to a randomly chosen corner of the data field. To terminate a trial, the participant had to press the blue button of the haptic device. This was only to be performed when the participant was convinced that the pointer location was at the point of interest.

Movements of the haptic device in 3D space as well as the time taken to complete each trial were recorded for analysis of speed and accuracy to find the correct point of interest within the field. Each experiment took between 30 to 60 minutes to complete.

5.3 Results

A total of 30 person participated in the studies, 10 participants in the frequency modulation, 10 in the positive amplitude modulation and 10 in the negative amplitude modulation condition. Each participant completed 36 trials (excluding the

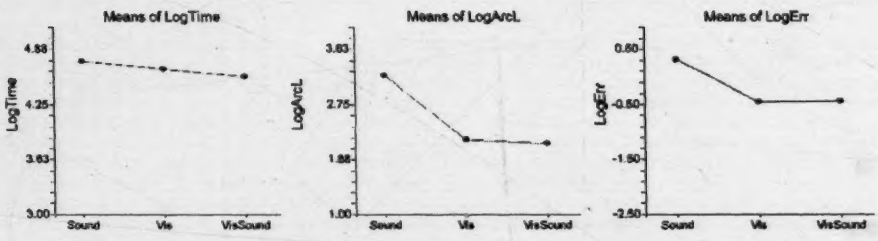


Figure 5.4: The results of trials with audio only vs trials with visual component

warmup trials) for a total of 1080 trials between all participants. Collected data for each trial consisted of time taken to complete the trial, the length of the path from the start to end point, the minimum distance between these two points, and the error distance between the selected and the correct end point. In addition, we also recorded information on the system's parameters for every trial.

All the data were analyzed using the NCSS program [14]. An outlier analysis was performed, and outlier data points were discarded. Only 1% of the data were considered to be outliers, while the rest 99% of the data was included in the analysis.

Given that the distributions of the dependent measures were highly skewed, we used log() as a normalizing transform for these measures. Appendix B shows some of these distributions before and after the normalization transform.

5.3.1 Sound Only vs Visual Component

To analyze how the audio-only feedback interface compares to the system setup with a visual component, we compared time (minutes), error and travel path (arc length) results for the corresponding dependant variables. Every participant had an equal number of audio only (Sound), visual only (Vis) and visual + audio (VisSound) trials. We conducted a within-subject ANOVA with factor VisSound (Sound, Vision, Sound + Vision). For all three measures, participants showed the worst results for the audio-only (Sound) system, which can be easily seen in Figure 5.4.

In the time analysis, log(time) showed a significant group effect [F(2,58)=21.23

for Log(time), $p<0.001$]. In addition, the Tukey-Kramer [5] test indicated that all three groups were different from each other. This indicates that participants were much slower in locating the goal with audio alone.

Similarly, for path length and error analysis, both measures showed significant group effects [$F(2,58)=327.97$ for Log(arc length), $p<0.001$; $F(2,58)=216.61$ for Log(error), $p<0.001$]. The Tukey-Kramer test indicates that the sound group is different from the other two groups. As one can see in Figure 5.4, participants were less efficient in exploring the volume in sound-only trials.

Effect of the Rendering Parameters on Sound Only Trials

Even though the audio-only feedback system was shown to be worse than the other two, it is still important to investigate how the other factors in the rendering algorithm compare. We conducted a within-subject ANOVA with factor Interpolation (Before, After) and factor Radius ($r=0$, $r=1.1$, $r=1.6$). We also conducted a between-subject ANOVA with factor Wave-modulation (frequency, positive amplitude, negative amplitude).

Interpolation did not have any effect on either time [$F(1,29)=0.09$, $p>0.05$ for Log(time)], path length [$F(1,29)=0.83$, $p>0.05$ for Log(arc length)], or error [$F(1,29)=0.42$, $p>0.05$ for Log(error)].

Similarly, Radius did not have any effect on either time [$F(2,58)=2.64$, $p>0.05$ for Log(time)], path length [$F(2,58)=2.52$, $p>0.05$ for Log(arc length)], or error [$F(2,58)=0.51$, $p>0.05$ for Log(error)].

Wave-modulation had no effect on path length [$F(2,347)=2.35$, $p>0.05$ for Log(arc length)], or time [$F(2,350)=1.31$, $p>0.05$ for Log(time)], but it had an effect on error [$F(2,347)=3,12$, $p<0.05$ for Log(error)], and Tukey-Kramer test indicated that positive amplitude modulation is significantly different from negative amplitude modulation for error. The corresponding error graph is shown in Figure 5.5. This

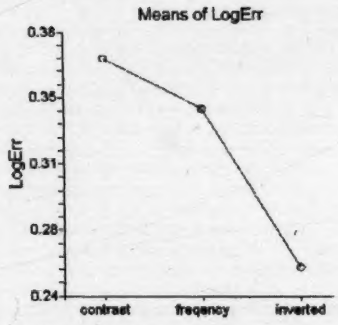


Figure 5.5: The results of wave-modulation factors on error in audio only trials

rendering parameters/ comparison criteria	*Time*	*Arc Length*	*Error*
interpolation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
radius	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
Wave modulation	no effect; p>0.05	no effect; p>0.05	some effect; p<0.05; negative amplitude
radius with Wave modulation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
radius with interpolation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
Wave modulation with interpolation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
All three groups of factors	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05

Table 5.1: Sound Summary Table: effects of different rendering parameters on sound only trials (shows p values for log measure, as well as the best setup if there is an effect)

information indicates that negative amplitude modulation helps to locate the goal position with the least error.

A mixed-factor ANOVA with factors Interpolation (Before, After), Radius (r=0, r=1.1, r=1.6) and Wave-modulation (frequency, positive amplitude, negative amplitude) showed that the interactions of either factors had no effect on either time, path length or error.

A summary of all the effects of different rendering parameters on sound only trials is presented in a Sound Summary Table (see Table 5.1).

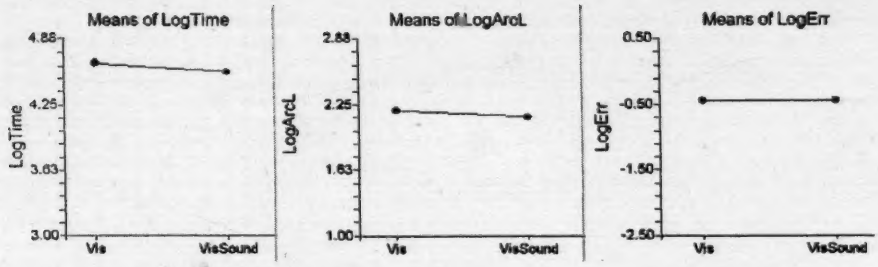


Figure 5.6: The results of multi-modal interface vs. visual only interface

5.3.2 Visual Only vs Visual + Sound

While the previous analysis mainly indicates the audio-only system setup to be worse than the other two, it does not say much about wether the multi-modal interface is any better than the visual only alone. Since the audio-only interface results were much different from the results for the other two conditions, they overshadowed the difference between the multi-modal and video-only interface. We thus decided to evaluate the results separately for the visual-only and visual+sound conditions.

Data analysis indicates that in all dependent measures the participants showed equal or better results for the multi-modal feedback system than for the visual only interface. This can be easily observed in Figure 5.6.

In time analysis, log(time) showed significant group effect [$F(1,29)=16.99$, $p<0.001$ for Log(time)]. In addition, the Tukey-Kramer test indicates that the visual condition is different from the visual + audio condition.

Regarding path length, there was a significant group effect as well [$F(1,29)=7.48$, $p<0.05$ for Log(arc length)]. We also analyzed a ratio of the path length to the minimum distance. Similarly to the path length analysis, log(ratio) showed significant group effects [$F(1,29)=4.77$, $p<0.05$ for Log(ratio)]. Tukey-Kramer test indicates that the visual group was different from the visual + audio group.

In the error analysis, no significant group effect was found [$F(1,29)=0.00$, $p>0.05$

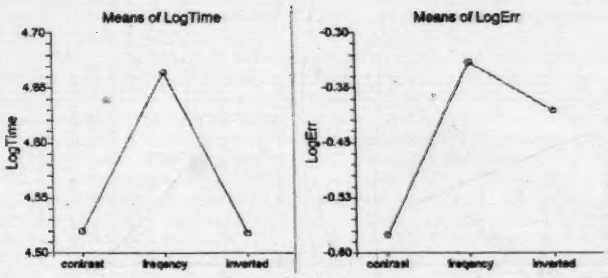


Figure 5.7: The results of wave-modulation schemes on time and error in visual + audio trials

for Log(error)].

Together with Figure 5.6, these results indicate that participants were much faster in locating the goal position and explored less space using a multi-modal (visual + audio feedback) system.

5.3.3 Effect of the Rendering Parameters in the Visual + Sound Condition

Sound parameters had effects in visual + audio trials that were similar to the ones shown for the audio-only trials in section 5.3.1.

Interpolation did not have any effect on either time [$F(1,29)=0.56$, $p>0.05$ for Log(time)], or path length [$F(1,29)=0.00$, $p>0.05$ for Log(arc length)], or error [$F(1,29)=0.78$, $p>0.05$ for Log(error)].

Similarly, Radius did not have any effect on either time [$F(2,58)=2.34$, $p>0.05$ for Log(time)], or path length [$F(2,58)=2.35$, $p>0.05$ for Log(arc length)], or error [$F(2,58)=0.38$, $p>0.05$ for Log(error)].

Wave-modulation had no effect on path length [$F(2,357)=2.85$, $p>0.05$ for Log(arc length)], but had a significant effect on time [$F(2,357)=10.05$, $p<0.001$ for Log(time)], and Tukey-Kramer test indicates that frequency modulation is different from both amplitude modulations schemes. Finally, Wave-modulation also effected error [$F(2,357)=6.59$, $p<0.01$ for Log(error)], and Tukey-Kramer test indicates that positive amplitude

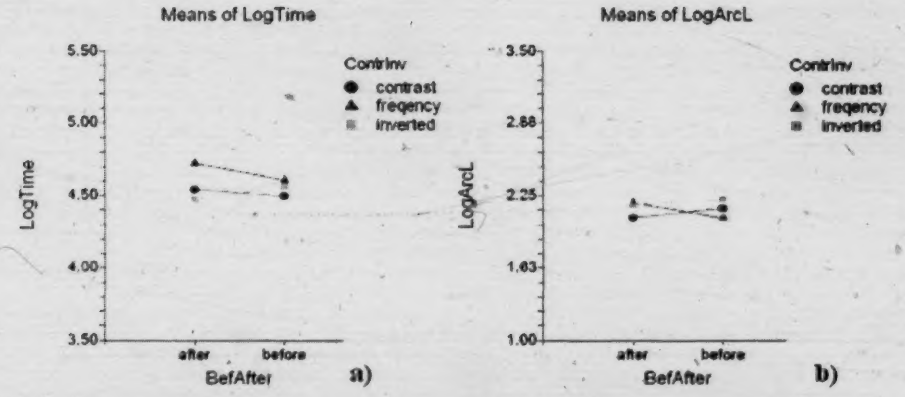


Figure 5.8: Interactions of different factors in visual + audio trials

modulation was different from the other two types. The corresponding time and error graphs are shown in Figure 5.7. This information indicates that amplitude wave modulation helps in navigating to the goal the fastest, while positive amplitude modulation helps to locate the goal with least error.

A mixed-factor ANOVA with factors Interpolation (Before, After), Radius (r=0, r=1.1, r=1.6) and Wave-modulation (frequency, positive amplitude, negative amplitude) revealed several significant interactions. Interaction of Radius with Wave-modulation, as well as Radius with Interpolation had no effect on either time, path length or error.

Interaction of Interpolation with Wave-modulation had no effect on error, but had some effect on time [$F(2,27)=4.39$, $p<0.05$ for Log(time)], and the Tukey-Kramer test indicates that frequency modulation after interpolation is different from both positive and negative amplitude modulation either before or after interpolation. Further, there was an effect on path length [$F(2,27)=6.28$, $p<0.01$ for Log(arc length)], and Tukey-Kramer indicates that negative amplitude modulation before interpolation is different from positive amplitude modulation after interpolation and frequency modulation before interpolation. The graphs in Figure 5.8 (a) and (b) indicate the behavioral change for different wave-modulation schemes in relation

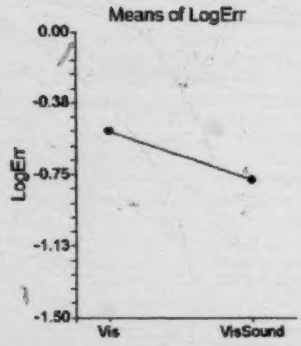


Figure 5.9: Multi-modal system in a specific setup vs. visual only system for error criteria

to Before-or-After interpolation schemes.

Interaction of all three factors show no effect on time or path length, but show significant effect on error [$F(4,54)=2.61$, $p<0.05$ for Log(error)]. This indicates that relation between different conditions influences the performance results. More precisely, a positive amplitude modulation before interpolation and largest radius show improvement over the visual-only trials with respect to error [$F(1,9)=5.64$, $p<0.05$ for Log(error)]. This can also be seen in Figure 5.9.

A summary of all the effects of different rendering parameters on visual+sound trials is presented in a Multi-Modal Summary Table (see Table 5.2).

5.4 Conclusion

This study was designed with several question in mind:

- How do visual-only, audio-only, and multi-modal feedback systems compare with respect to time, path length and accuracy?
- What are the best audio rendering parameters?
- Does the same audio setup work best for both audio-only and multi-modal system with respect to all measures?

rendering parameters/ comparison criteria	*Time*	*Arc Length*	*Error*
interpolation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
radius	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
Wave modulation	significant effect; p<0.001; amplitude	no effect; p>0.05	significant effect; p<0.01; positive amplitude
radius with Wave modulation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
radius with interpolation	no effect; p>0.05	no effect; p>0.05	no effect; p>0.05
Wave modulation with interpolation	some effect; p<0.05; behavioral change	significant effect; p<0.01; behavioral change	no effect; p>0.05
All three groups of factors	no effect; p>0.05	no effect; p>0.05	some effect; p<0.05; r=1.6 with positive amplitude and before interpolation

Table 5.2: Multi-Modal Summary Table: effects of different rendering parameters on sound+visual trials (shows p values for measure and log measure, as well as the best setup if there is an effect)

- Does the same audio setup work best for all measures?

The results indicate that the audio-only feedback system produces the worst performance, while multi-modal feedback system shows an improvement in localization over the visual-only system with respect to most measures.

5.4.1 Drawbacks of the Audio-only system

The reason for such poor performance under audio-only system with respect to both time and path length is that the person initially does not have any indication of where the goal is, which means that one has to blindly explore a large portion of the volume before discovering the area of interest. This takes a lot of movements and time, and even after the general area of interest is found, the precise location of the goal requires many small movements in all the possible directions to pinpoint the exact location of the vortex center.

The reason for poor accuracy in the audio-only condition is not straight forward. Taking into account the data-to-sound transformation function, it becomes clear that the area near the goal might form a plateau in sound values, making it hard

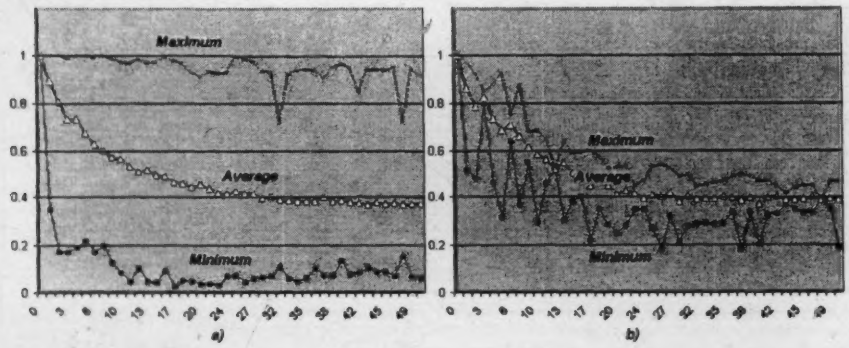


Figure 5.10: Sound Loudness (Y axis) vs. Square Distance (X axis) from the goal, showing minimum, maximum and average loudness values in that distance. a) Average over all 36 fields; b) One of the fields

to pinpoint the exact goal location (see Figure 5.10a). There is also a high chance of false-positives when the participant hits a local maximum (see Figure 5.10b for an example field with local maximum) either because he is sure to be at the global maximum and/or is unwilling to move and explore more of the local area for a fear of loosing the position found. Also, since audio-only trials take the most time for exploring, by the time the general area of interest is found, a fatigue factor might become important.

5.4.2 Advantages of The Multi-Modal Interface

As expected, the multi-modal interface gives the best results over the other two systems.

First of all, from the visual clues the user knows where the goal is. This is an advantage over the audio-only system, since it takes less movements and time to get to the area of interest as the visual component clearly indicates where the global maximum is, reducing the navigation path length and time. It also reduces the chance for false positives. Visual false positives, resulting from the 2D view of 3D world, are reduced with audio cues. Audio false positives, resulting from the local

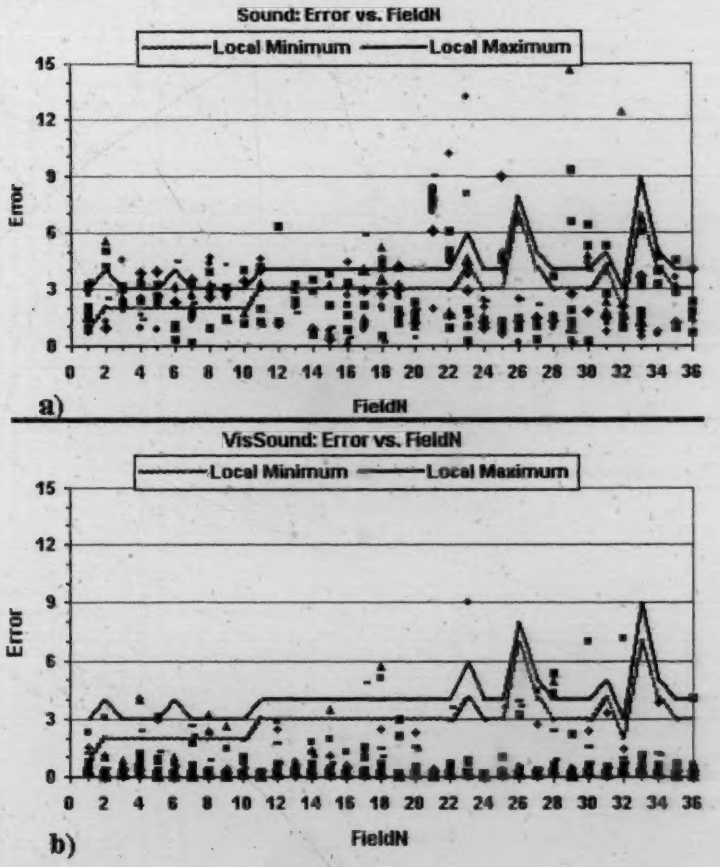


Figure 5.11: Error Distance vs. Field Number, showing closest to the goal local minimum and local maximum for every field a) Error distribution in the case of audio-only interface; b) Error distribution in the case of the multi-modal interface

maximums, are reduced with visual cues. This scenario is illustrated in Figure 5.11. As shown, in multi-modal case (Figure 5.11b), the error distance for most responses is closer to the goal than either closest local maximum or local minimum. Audio-only interface (Figure 5.11a) on the other hand, shows a lot of responses around the distance regions of the closest local maximum.

Further, the visual clues only give the user a 2D view of the 3D world on the computer screen. Audio clues add a third dimension. This l ts the user align the

cursor with the goal in the 2D view of the screen and then move it in and out of the screen while listening to audio clues to align it with the goal position in 3D. Another technique that one can use is to roughly locate the area of interest and then use the audio clues to precisely locate the goal position. All of this reduces the navigation time, since the user does not have to spend as much time rotating the volume to locate the correct 3D location of the goal. It also reduces path length, since reliance on the 2D view alone might result in many movements in wrong directions. Audio clues add information on whether one is moving towards or away from the goal in a local area. This in turn, also helps to reduce the time taken to locate the goal position.

The reason that, in many cases, the multi-modal system is only as good as the visual-only system with respect to accuracy is related to the possible sound plateau around the goal, as was the case in the audio-only system (see Figure 5.10). False positives and fatigue do not have an effect in this case, and a proper audio setup that reduces the plateau effect can improve performance significantly. This possibility is obvious since in the case where the multi-modal interface parameters are set to positive amplitude modulation before interpolation setup and a largest radius of influence actually shows some improvement over the visual-only condition with respect to accuracy.

5.4.3 Audio Rendering Parameters

Both, audio-only and multi-modal interface can be improved with a specific set of audio rendering parameters. None of the audio parameters used make audio-only system better than either the visual-only or the multi-modal system. However, some parameter values do give better results than others. Negative amplitude modulation improves user accuracy in the audio-only system. This specific rendering mode helps to reduce the number of false positives since the user tries to first find a silent

spot in the local area, and, if none is found, resorts to looking for the quietest one. Locating either the loudest sound or the highest frequency sound may be more ambiguous.

In the multi-modal interface, specific audio setups also give better performance. Amplitude wave modulation improves performance with respect to time, while positive amplitude improves performance with respect to accuracy. Notice that this is different from the audio-only system. This can be explained by the addition of a visual component to the system. Thus, finding a general area of interest and reducing false positives is mainly due to visual clues. After the general area is located, positive amplitude changes near the vortex center help to locate the goal better than negative amplitude changes.

While some setups of the interface clearly indicate improvement in performance, some other ones show how different factors interact with each other. As shown in Figure 5.8, different wave-modulation factors influence the effectiveness of Before-or-After interpolation. Also, as mentioned previously, a specific sound rendering setup, namely the positive amplitude modulation with the largest radius before interpolation, makes the multi-modal interface better than the visual-only system with respect to accuracy. This shows that a careful selection of the audio parameters and rendering schemes can greatly improve the performance of the interface.

In conclusion, we find that the multi-modal interface with video and sound behaves significantly better than sound or video alone for this localization task; second that there are specific sets of audio rendering parameters and schemes that can really improve the performance of the interface; third that there are different audio parameters that are better for audio-only than for the multi-modal interface; and finally that audio parameters may have to be different for different conditions, although a compromise can be achieved.

Chapter 6

Conclusion

This thesis presents a multi-modal interface that uses sonification alongside with visualization to render complex CFD data sets. In the thesis, we discuss how different types of sonifications can be used to render the complex CFD data. The different sonifications were then studied experimentally. Results showed several advantages of the multi-modal interface over pure visualization. The experimental study also helped determine how some sonification mappings are better at helping users to localize vortices in CFD data.

6.1 Summary

In this thesis, we discuss the problems associated with visualization of high dimension complex CFD data sets. We mention the problem of dimensionality curse and the various problems associated with visual cluttering of data representation suiting conventional scientific visualization schemes. In this context, we were able to demonstrate that sonification can help to solve some of these problems. Various types of CFD specific visualization techniques, such as different types of complex glyphs and hyperstreamlines, still leave many problems open. Sonification can be used to try to tackle some of them. In short, sonification can help reinforce the clues given through other senses; it can help recognize the features not obvious from other sources; it can help reduce the dimensionality curse by representing different data

attributes to different senses.

To demonstrate these concepts in practise, we implemented and presented a multi-modal visual/audio interface for interacting with CFD data. To prove the advantages of this system in practise, we conducted a set of user studies that showed experimentally the advantages of this interface.

6.1.1 Multi-Modal System Summary

The multi-modal interface developed runs on separate threads for visualization and sonification, thus allowing both types of rendering to get as complex as necessary without interfering with each other. Both processes read the same data set and receive the same interaction events at every moment in time, such as the new pointer position and direction in 3-D space, and they thus are completely synchronized. Haptics was mainly used for indicating the virtual pointer position and orientation and for other minor tasks such as defining the boundary of the flow field.

As discussed in Chapter 4, there are endless options on how to map data to sound. The two main objectives of our mapping algorithms were 1) that they had to be psycho-physically justified and 2) that they were easy to understand. The produced sound had to represent changes in data values as distinctly and accurately as possible, while being pleasant and easy to listen to.

We developed several types of mappings to see if different types of sonification influence the effectiveness of the system. All of these mappings used broad band noise, and we modified some of its characteristics to represent the CFD information. These characteristics included the number of sound waves produced (after interpolation had one sound wave characterizing all the points, before interpolation produced sound waves for each data point), the number of data points influencing the mapping, as well as the type of wave modulation portraying the changes in data values (amplitude vs. frequency modulation, positive vs. negative mapping).

To prove the concepts, we chose fluid field velocity values of a given CFD data set for visual and audio rendering and conducted a set of experiments to see how users performed with our multi-modal interface. Although both, sonification and visualization programs can potentially be very complex, we used fairly simple types of visual and audio renderings for comparing visual only, sound only and multi-modal systems.

6.1.2 Usability Study Summary

The most significant result of our experiments is multi-modal systems perform better than either pure visual or pure audio systems.

Although the audio-only system gave the worst results, experiments showed that specific audio configurations can help to improve performance for both, multi-modal and audio-only systems.

We analyzed the time it took the user to move to a specific location within the virtual fluid field, the length of the travel path, and the accuracy in localizing the goal. Multi-modal system showed the best results with respect to time and path length, and in certain cases it also gave the best accuracy performance.

The user studies also showed that the effectiveness of audio setup depended on the existence of visual clues as well as on the specific criteria being tested and on the individual test.

Overall the results showed that:

- Multi-modal systems are more powerful than either pure visual or pure audio systems;
- Specific mapping parameters do influence system performance;
- It is important to re-evaluate the system for better performance if more components are added;

- There are many possibilities for creating a very helpful and powerful sonification tool that can be modified in correspondence to specific needs of the system.

Results showed how sonification can help to improve perception in multi-modal system and to provide more confidence in integrating other sensing modalities, such as haptics, to assist in analyzing complex CFD data sets.

6.2 Future Development

There are a lot of options for further development and improvement of the multi-modal interface. More sophisticated rendering algorithms can be used for visualization, sonification and haptic representation of the data. In addition, an overall system can be made more flexible to tailor to the needs and preferences of the user.

6.2.1 Improvements and Other System Setups

One of the major system improvements would be to add a more sophisticated haptic program to the system. There are a lot of different possibilities in this case. First, different types of algorithms can be chosen to do haptic rendering of the data. This might depend on the data type being rendered, the required complexity of the output, and the type of output the visual and audio components of the system are using. In each cases, a user study needs to be performed to see how the haptic component influences performance and to determine which haptic rendering algorithm and parameters are the best. It is our experience, ad-hoc definitions of these parameters are never good.

There are always more opportunities for creating other audio setups and testing their performance with the system. One of the possibilities is to step away from the linear mapping functions for mapping data values to sound properties. Instead, a polynomial or exponential function can be used to intensify sounds near the critical

values. This might improve system performance, especially with respect to accuracy, since sound around the critical value would be more informative. Sonification can also be done along path lines, streamlines, streamribbons and streamtubes (see section 2.2.3). Another set of user studies will be required to confirm that these modifications improve usability significantly.

Not only can different mapping functions be used, but other data properties can be sonified using similar or completely different types of data-to-sound mappings. Several data properties of the complex tensor of the fluid field can be sonified at once. In this case, a sound mapping function should be complex enough and at the same time easily interpretable, so that each component of the complex tensor is easily identifiable in the synthesized sound.

Another option would be to sonify only one data property at a time, but give the user an option to switch between sonifying different data properties, as needed during the exploration of the data field.

Another major system improvement would be to create a fully spatialized surround-sound using several speakers around the user's head. This would help immerse the user into the virtual fluid field by giving him a better sense of localization. An early experiment on this form of rendering was tested by Taylor [29]and a usability study is currently under development to evaluate the improved efficiency. Finally, visualization can also be done through more immersive large 3–D displays instead of a small computer screen.

6.2.2 Self-Adjustable system

An interesting trend, noticed during the user studies, is that different people may have different preferences regarding the audio setup. This trend was not analyzed in detail in this thesis, as it was not in the framework of the current research. However, it might be interesting to first determine the preferred audio setup for each user,

before letting them use the system to its full capacity.

The system can run one specific setup for general use but have an option for adapting to a more specialized user. An information log can be kept in the system with information of the preferred setups for each user. When necessary (if the user wants to, or some property of the system was modified) another user-system test should be performed and the user information in the system updated.

6.3 Possible Applications

There are many possible applications for the sonification field as a whole and for CFD sonification in particular. One of the applications that CFD field sonification is directly applicable, is to study air and liquid motion, simulating such processes as winds over the terrain and air or liquid flow around large and/or complex objects.

We created our sonification program to be incorporated in the CFD simulation solution system developed by Boulanger et al. [2]. In this system, the simulation server is run on a high performance computer and is responsible for all the necessary computations of CFD simulation and time-step calculations, which are then distributed in real-time to the client nodes. Client programs are mostly responsible for displaying the received information to the user, which can be done through visual, audio and/or other types of rendering. The user can then interact with the simulation by sending different types of requests to the server and receiving updated information in real-time. In this system, data computation is separated from the rendering and interaction, which allows for separate implementation of the necessary rendering programs, which can be visual, audio or multi-modal or some other configuration. Client programs could be adjusted according to the user's needs to run the preferred rendering/interaction setup. This would also allow client programs to be developed at their own pace and complexity separately from the server.

In our multi-modal client program, both visualization and sonification of the

data are given to the user, with haptic interface allowing for interaction with the dataset. In the current setup, both visualization and sonification processes receive the whole data set at each time step from the server as it is being computed. Sonification is then done locally. In a more complex setup, a significant part of the sonification process can be moved over to the server. As shown in Figure 1.1, information for the visual display can be given to the client as an image or geometry or a mixture of both. Similarly, audio information can be given to the client as a sound wave data or geometry (either of the whole field or of the local area of interaction only), or some mixture of both.

6.3.1 Virtual Wind Tunnel

Wind tunnels are essential for studying different phenomena. However, they are not always accessible. Computational fluid dynamics is a great tool for simulating fluid flow. Thus, it is heavily used in designing aircrafts, cars and other prototypes that depend on interaction with air and fluid flows.

In our simulation of the 'Virtual Wind Tunnel', the user is able to interact with the virtual fluid flow through visual, sound and haptic interfaces, that allow the user to move to a different position in the flow while observing and listening to the complex processes of the fluid interacting with various virtual objects or terrains.

As was shown in our user study, a multi-modal interface allows for better interaction between the user and the system that is rendering CFD fluid flow data computed by our solution server. Thus, a multi-modal 'Virtual Wind Tunnel' gives the user a better understanding of the complex processes of the simulated fluid field.

6.3.2 Real-time Architecture

In our simulation system of the 'Virtual Wind Tunnel', interaction with the flow is done in real-time. Not only can the user navigate within the flow in real-time while viewing and listening to local subset of data, but the changes to the simulation

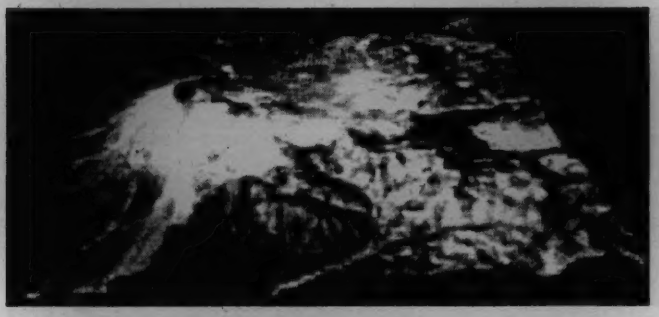


Figure 6.1: Terrain model of Mount Saint Helens

process can also be done in real-time. Thus the user can modify some simulation parameters and see the changes in the solution data update with these modifications while the program is running.

In the multi-modal system this would allow the user to see and evaluate the current output of the solution process through visual, audio, haptic, and potentially other cues. The combination of these outputs can help the user to better understand the given CFD solution, and to understand changes to the solution if the user chooses to modify the simulation parameters. This way, when the parameters are modified, not only will the visual display of the fluid data set change, but also the sound will change as well. This might help the user to notice and understand the changes in the solution.

6.3.3 Wind Studies Over a Terrain Model

We used a digital terrain model of Mount Saint Helens as a test bed for the multi-modal CFD solution system (see Figure 6.1). The conversion of the mesh to the CFD domain is described in detail in [2]. The generated CFD grid represents the air flow over the mountain surface, and describes a space that is 18838 metres by 17476 metres by 6000 metres large. The solution generated by the solution server outputs 14222 nodes, with each data point containing information describing fluid flow velocity, pressure and other values at a given position of the flow. A rendering

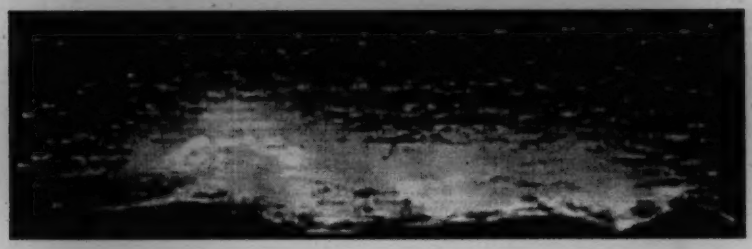


Figure 6.2: Velocity vectors over Mount Saint Helens for a laminar wind

of the velocity vectors over the mountain surface is shown in Figure 6.2.

Several time-steps for the low altitude winds over the mountain were generated. Multi-modal visulaization/sonification program could then render any of the given time-steps, while allowing the user to move inside the data set using a haptic interface. To give the user more options, the digital model of the Mount Saint Helens could be switched on and off. Sonification was done using either rendering of the audio setups as described in the Section 4.3, with the user having control over the local area radius. Then the radius could be made larger for a general exploration of the volume and reduced in the areas of interest for a more local exploration.

6.4 Final Remarks

In this thesis, we presented sonification as a part of the multi-modal interface for CFD fluid field data analysis. A visual/audio multi-modal system was developed with several types of mappings for mapping CFD data to sound. We chose fluid velocity (which is one the major fluid flow characteristics) for sound rendering. To simulate the sound of the wind, we used broad band white noise with varying average amplitude and central frequency. Data-to-sound mappings were chosen to enhance human perception. Thus, we tried to achieve linear relationship between flow velocity and loudness of the noise.

Through a set of user studies we showed that sonification does help improve perception. Sonification alone gave poor results. However, a multi-modal vi-

sual/auditory system gave the best results in comparison to the single-modality systems, showing that addition of sonification to visualization system helps to improve performance. We also showed how different types of sonification algorithms influence performance, and how they can help to improve CFD data analysis.

The user study also showed that different types of sonification algorithms influence multi-modal and pure audio systems differently. This demonstrates the importance of system evaluation with addition of new components.

At the end, we briefly presented a possible application for the proposed sonification. In our 'Virtual Wind Tunnel' system, multi-modal implementation widened the ways for the user to observe and interact with the virtual wind flow, thus enhancing the application and making it more versatile.

Overall, we showed the advantages of using multi-modal system for analyzing CFD fluid field data. This research can potentially contribute to the advancement of our understanding of how auditory and multi-modal interface can improve human exploration skills and assist with real-time CFD analysis.

Bibliography

[1] M. Ballora, B. Pennycook, P.C. Ivanov, L.Glass, and A.L. Goldberger. Heart rate sonification: A new approach to medical diagnosis. *LEONARDO*, 37(1):41–46, 2004.

[2] P. Boulanger, M. Garcia, C. Badke, and J. Ryan. An advanced collaborative infrastructure for the real-time computational steering of large cfd simulations, September 2006.

[3] T. Bovermann, T. Hermann, and H. Ritter. The local heat exploration model for interactive sonification. In *Proceedings of International Conference on Auditory Display*, pages 85–91, July 2005.

[4] E. Childs. The sonification of numerical fluid flow simulations. In *Proceedings of the 2001 International Conference on Auditory Display*, pages 44–49, July 29 - August 1 2001.

[5] C. E. Collyer and J. T. Enns. *Analysis of Varience The Basic Designs*. Chicago: Nelson-Hall Publishers, 1986.

[6] Cycling '74. Max/MSP, 2004.

[7] T. Delmarcelle and L. Hesselink. Visualizing second-order tensor fields with hyperstreamlines. *IEEE Computer Graphics and Applications*, 13(4):25–33, July 1993.

[8] EVL Electronic Visualization Laboratory. QUANTA, July 2002.

[9] T.G. Farr and M. Kobrick. Shuttle radar topography mission produces a wealth of data. *Amer. Geophys. Union Eos*, page 81:583585, 2000.

[10] S.A. Gelfand. *Hearing: An Introduction to Psychological and Physiological Acoustics, Forth Edition, Revised and Expanded.* New York: Marcel Dekker, 2004.

[11] B. Gold and N. Morgan. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music.* New York: Wiley, 2000.

[12] W.L. Gulick, G.A. Gescheider, and R.D. Frisina. *Hearing: Physiological Acoustics, Neural Coding, and Psychoacoustics*. Oxford: Oxford University Press, 1989.

[13] Y. M. A. Hashash, J. I. Yao, and D. C. Wotring. Glyph and hyperstreamline representation of stress and strain tensors and material constitutive response. *International journal for numerical and analytical methods in geomechanics*, 27(7):603–626, 2003.

[14] J. Hintze. ncss, 1981.

[15] S. Ishihara. *Tests for colour-blindness*. Tokyo, Kanehara Shuppan Co., 1978.

[16] H.G. Kaper, S. Tipei, and E. Wiebel. Data sonification and sound visualization. *Computing in Science and Engineering*, 1(4):48–58, July - August 1999.

[17] Kitware. VTK.

[18] E. Klein and O. G. Staadt. Sonification of three-dimensional vector fields. In *Proceedings of the SCS High Performance Computing Symposium*, page 8, 2004.

[19] K. Metze, R.L. Adam, and N.J. Leite. Cell music: The sonification of digitalized fast-fourier transformed microscopic image. In *Proceedings of XXIII Congress of the Brazilian Society of Computation*, pages 119–124, 2003.

[20] M. Noirhomme-Fraiture, O. Scholler, C. Demoulin, and S. Simoff. Sonification of time dependent data. In *Proceedings of International Workshop on Visual Data Mining*, pages 113–125, August 2002.

[21] OpenCFD. OpenFoam, 2004.

[22] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit An Object-Oriented Approach To 3D Graphics, 3rd Edition*. Kitware, Inc. publishers, 2003.

[23] Sensable Tech. Inc. PHANTOM Omni.

[24] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference*, page 517524, 1968.

[25] Y. Shin and C. Bajaj. Auralization i: Vortex sound synthesis. In *Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization*, 2004.

[26] Silicon Graphics, Inc. (SGI). OpenGL Performer, 1991.

[27] C. Spence and J. Driver. *Understanding intersensory integration*. Oxford: Oxford University Press, 2004.

[28] S.S. Stevens. The direct estimation of sensory magnitudes: loudness. *The American Journal of Psychology*, 69(1):1–25, March 1956.

[29] R. Taylor, M. Kazakevich, P. Boulanger, M. Garcia, and W. F. Bischof. Multimodal interface for fluid dynamics simulations using 3-d localized sound. In *In Proceedings of 7th International Symposium on Smart Graphics*, pages 6 pages, in press, June 25-27 2007.

[30] Russell M. Taylor II, Thomas C. Hudson, Adam Seeger, Hans Weber, Jeffrey Juliano, and Aron T. Helser. VRPN: A device-independent, network-transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 55–61. ACM Press, 2001.

[31] National Geospatial-Intelligence Agency (NGA) the National Aeronautics and Space Administration (NASA). The shuttle radar topography mission (srtm), 2000.

[32] G. Tzabiras. *Calculation of complex turbulent flows*. Boston : WIT Press, 2000.

[33] B.N. Walker. *Magnitude Estimation of Conceptual Data Dimensions for Use in Sonification*. PhD thesis, Rice University, 2000.

[34] N. Walker and J.T. Cothran. Sonification sandbox: a graphical toolkit for auditory graphs. In *Proceedings of the 9th International Conference on Auditory Display*, pages 161–163, July 2003.

[35] R.M. Warren. *Auditory Perception: A New Analysis and Synthesis*. Cambridge University Press, 1999.

[36] R. B. Welch and D. H. Warren. Intersensory interactions. In K. R. Boff, L. Kaufman, and J. P. Thomas, editors, *Handbook of Perception and Human Performance: Vol 1. Sensory Processes and Perception*, pages 25–1 – 25–36. New Cork: Wiley, 1986.

[37] W.A. Yost. *Fundamentals of Hearing: An Introduction, Forth Edition*. New York: Academic Press, 2000.

[38] J. Zhong, J. Weng, and T.S. Huang. Robust and physically-constrained interpolation of fluid flow fields. In *ICASSP'92: Acoustics, Speech and Signal Processing Conference*, volume 3, pages 185–188, March 1992.

Appendix A

User Studies Instructions

You will be presented with a virtual field of data and asked to find a specific point of interest that is indicated either by the largest arrow or loudest sound or both. The **loudest** sound is always associated with the largest **red** arrow, whereas quiet sound is always associated with small, light blue arrows.

Point of interest is always at the **beginning** of the red arrow.

To hit the point of interest with a pointer, the two arrows have to **coincide** at their **starting points**.

The top right front corner of the data cube is identified by the little blue sphere attached to it. This way when the cube is rotated you still know where the front, top, etc is. You can rotate the cube in the visual window using the **middle mouse button** (click **F1** to go to the original orientation). However, that does not have a real affect on the cube's orientation, for the haptic device it always stays the same.

If the pointer arrow (which is always white) is occluded by the point of interest arrow, then pointer arrow is behind. If, otherwise, a point of interest arrow is occluded by the pointer arrow, then the pointer arrow is in front. This way you can use occlusion to navigate.
If the pointer arrow disappears out of the view during a visual trial, it is most probably behind the cube. Similarly, when the sound disappears during the Sound trial, you are most probably pointing outside the cube.

The experiment is divided in two parts. In 15 warm-up trials, you are encouraged to play the interface, try rotating the cube and experiment with navigating within it. This is followed by an experimental set consisting of 36 trials. For each trial, you will be presented with one randomly chosen field and randomly chosen Condition. A condition might be either Visual only, Sound only or Visual + Sound. The middle screen shows the data field, and the left screen informs you about the trial that is being presented. All fields have the some dimensions, but the point of interest is at different locations.

Please, take a brief break between the two parts.

To **start the next trial** you should click only the **white** button on the haptic device. To **finish the trial**, you should click only the **blue** button on the haptic device. This should be done only when you think you are at the point of interest. Each new trial will start off with a haptic pointer at a randomly chosen corner of field data cube. After each trial, allow the haptic device to navigate you to the next randomly chosen corner by holding it lightly. Stay at that corner until you start the next trial.

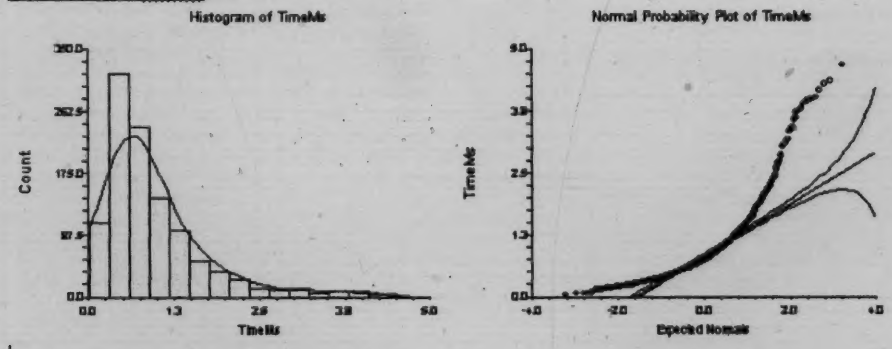
Thank you for your participation.

Appendix B

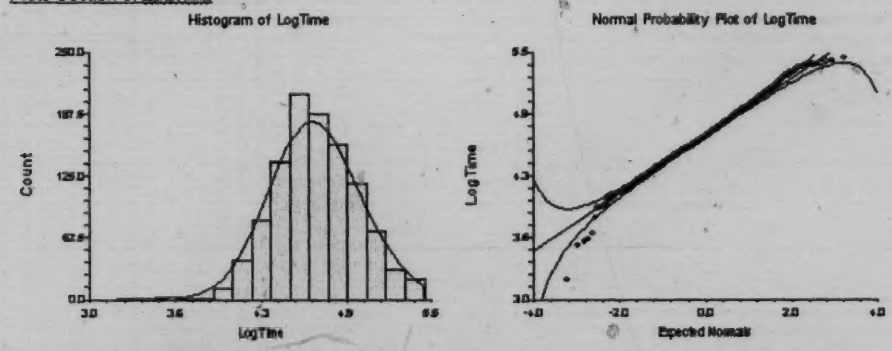
Measure and Log(Measure) graphs

B.1 Time vs. Log(Time)

Plots Section of TimeMs



Plots Section of LogTime



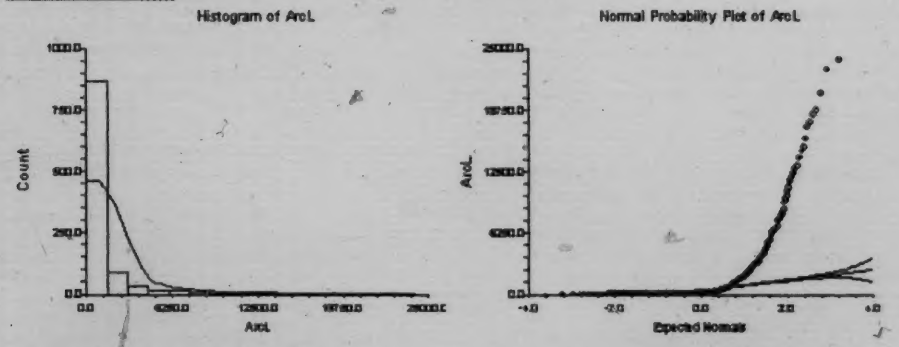
For time: the values for Skewness and Kurtosis are 2.00 and 7.75 respectively; while all the Normality Tests reject normality.

For Log(Time): the values for Skewness and Kurtosis are -0.06 and 3.50 respectively; while Normality Tests rejecting normality are Shapiro-Wilk W, Martinez-

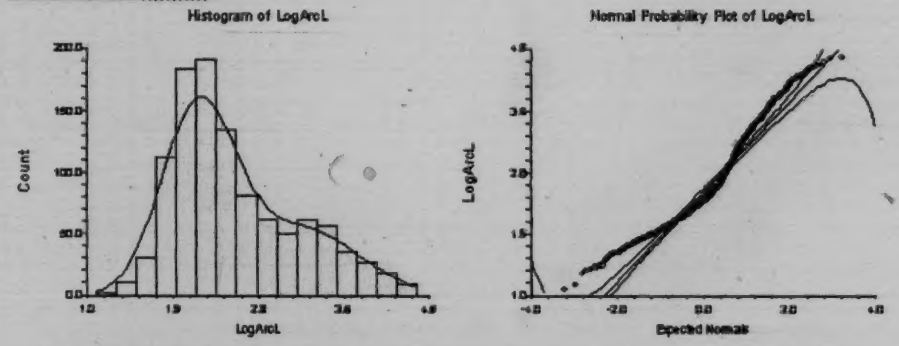
Iglewicz, D'Agostino Kurtosis and D'Agostino Omnibus; and Normality Tests that cannot rejecting normality are Anderson-Darling, Kolmogorov-Smirnov and D'Agostino Skewness.

B.2 ArcLength vs. Log(ArcLength)

Plots Section of ArcL



Plots Section of LogArcL

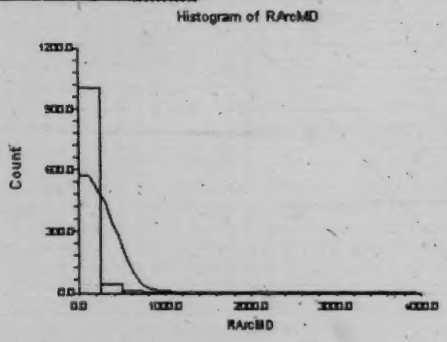


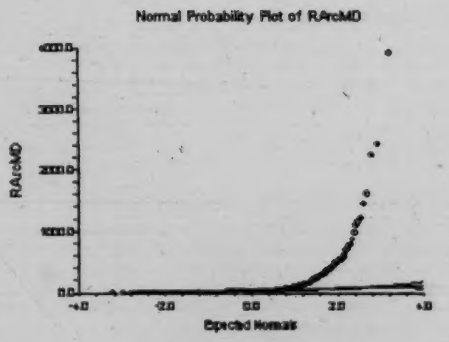
For arcLength: the values for Skewness and Kurtosis are 4.42 and 26.86 respectively; while all the Normality Tests reject normality.

For Log(arcLength): the values for Skewness and Kurtosis are 0.79 and 2.87 respectively; while Normality Tests rejecting normality are Shapiro-Wilk W, Martinez-Iglewicz, D'Agostino Omnibu, Anderson-Darling, Kolmogorov-Smirnov and D'Agostino Skewnesss; and Normality Test that cannot rejecting normality is D'Agostino Kurtosis.

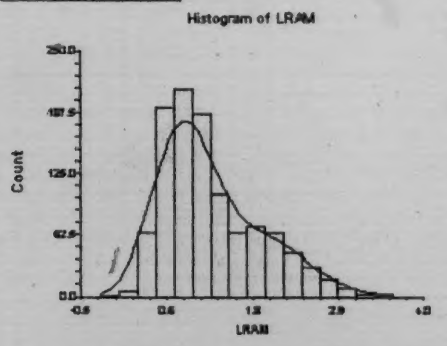
B.3 Ratio vs. Log(Ratio)

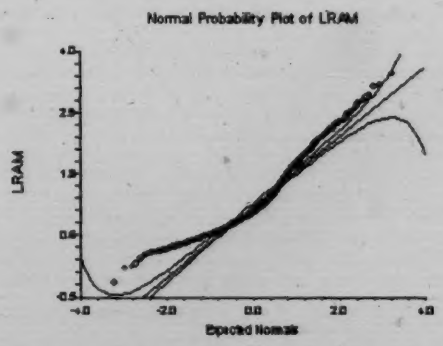
Plots Section of RArcMD





Plots Section of LRAM



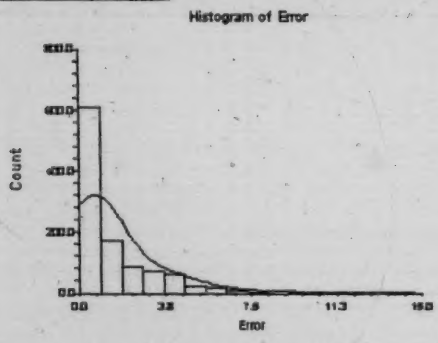


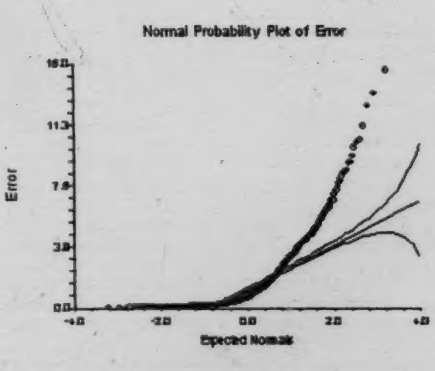
For ratio of arcLength to the minimum distance: the values for Skewness and Kurtosis are 10.20 and 26.86 respectively; while all the Normality Tests reject normality.

For Log(ratio): the values for Skewness and Kurtosis are 0.92 and 3.25 respectively; while Normality Tests rejecting normality are Shapiro-Wilk W, Martinez-Iglewicz, D'Agostino Omnibu, Anderson-Darling, Kolmogorov-Smirnov and D'Agostino Skewnesss; and Normality Test that cannot rejecting normality is D'Agostino Kurtosis.

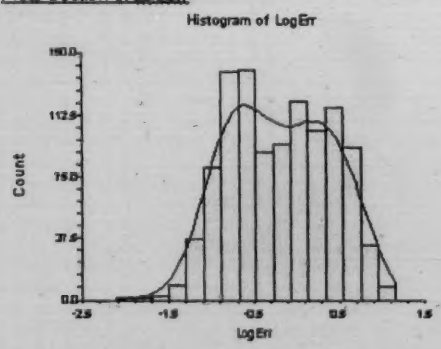
B.4 Error vs. Log(Error)

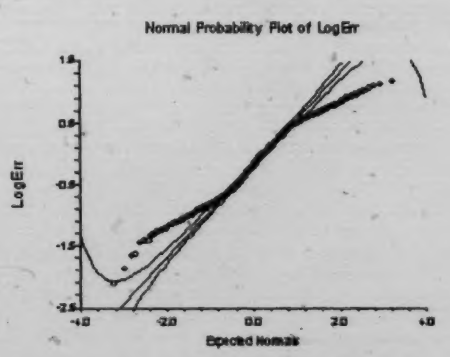
Plots Section of Error





Plots Section of LogErr



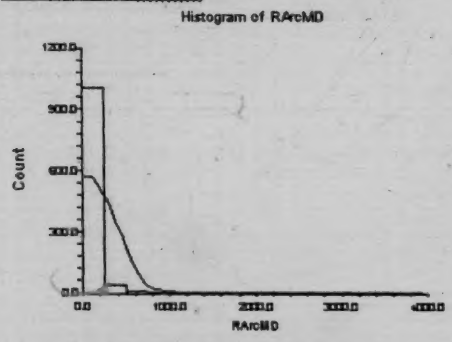


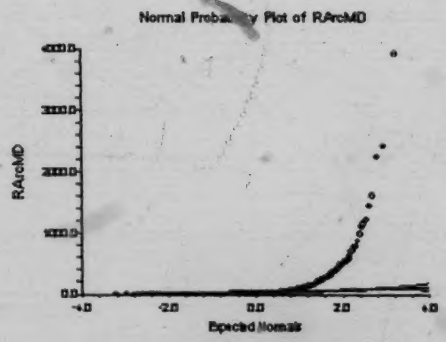
For error: the values for Skewness and Kurtosis are 2.35 and 10.57 respectively; while all the Normality Tests reject normality.

For Log(error): the values for Skewness and Kurtosis are 0.01 and 2.07 respectively; while Normality Tests rejecting normality are Shapiro-Wilk W, D'Agostino Omnibu, Anderson-Darling, Kolmogorov-Smirnov and D'Agostino Kurtosis; and Normality Tests that cannot rejecting normality are Martinez-Iglewicz and D'Agostino Skewness.

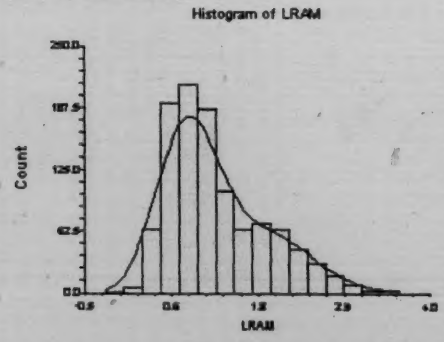
B.3 Ratio vs. Log(Ratio)

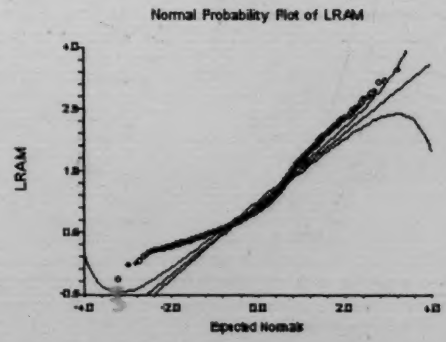
Plots Section of RArcMD





Plots Section of LRAM



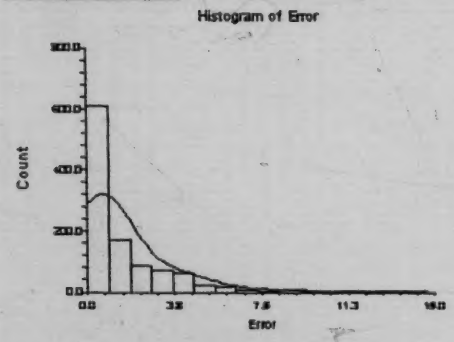


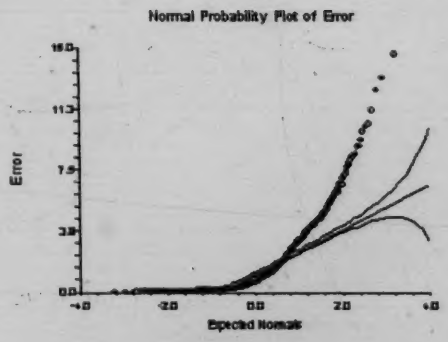
For ratio of arcLength to the minimum distance: the values for Skewness and Kurtosis are 10.20 and 26.86 respectively; while all the Normality Tests reject normality.

For Log(ratio): the values for Skewness and Kurtosis are 0.92 and 3.25 respectively; while Normality Tests rejecting normality are Shapiro-Wilk W, Martinez-Iglewicz, D'Agostino Omnibu, Anderson-Darling, Kolmogorov-Smirnov and D'Agostino Skewnesss; and Normality Test that cannot rejecting normality is D'Agostino Kurtosis.

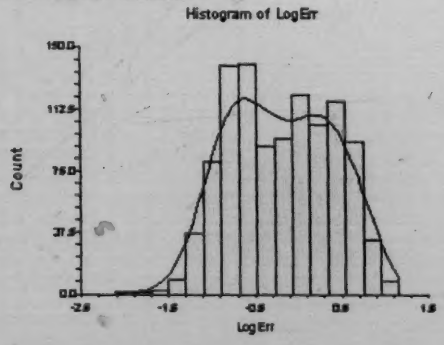
B.4 Error vs. Log(Error)

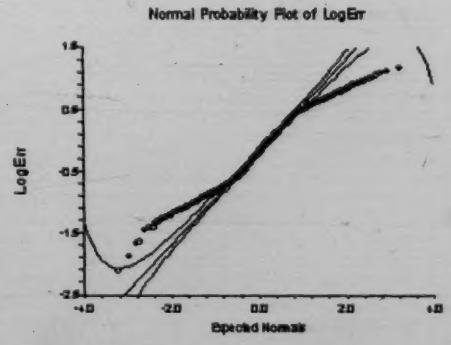
Plots Section of Error





Plots Section of LogErr





For error: the values for Skewness and Kurtosis are 2.35 and 10.57 respectively; while all the Normality Tests reject normality.

For Log(error): the values for Skewness and Kurtosis are 0.01 and 2.07 respectively; while Normality Tests rejecting normality are Shapiro-Wilk W, D'Agostino Omnibu, Anderson-Darling, Kolmogorov-Smirnov and D'Agostino Kurtosis; and Normality Tests that cannot rejecting normality are Martinez-Iglewicz and D'Agostino Skewness.